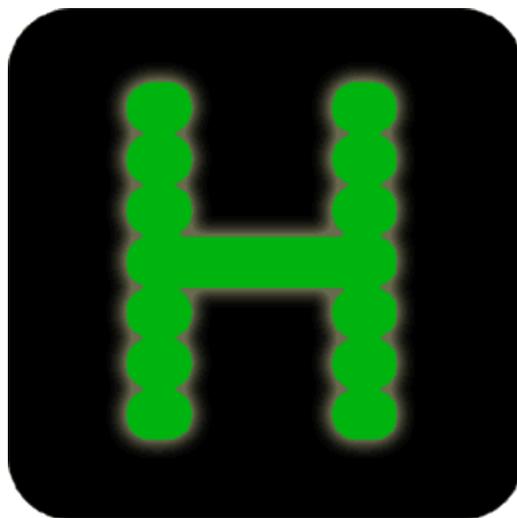**Hercules System/370, ESA/390, z/Architecture Emulator**

# Hercules – Installation Guide
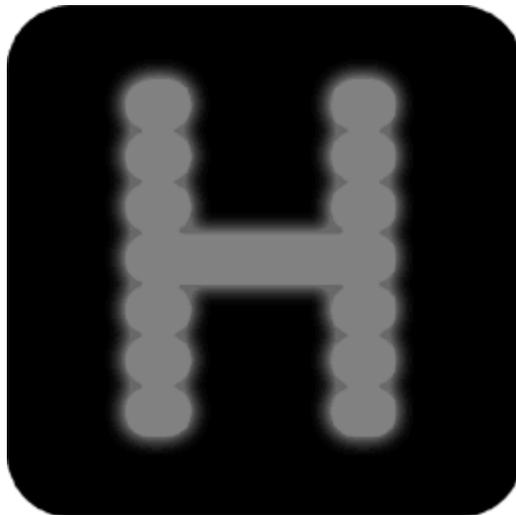
*Version 3  Release 09*

**Hercules System/370, ESA/390, z/Architecture Emulator**

# Hercules – Installation Guide

*Version 3  Release 09*



**First Edition, July 15, 2013**

**HEIG030900-00**

# Contents

# Figures

# Tables

# 1. Preface

## 1.1 Edition information

This edition applies to the Hercules S/370, ESA/390 and z/Architecture Emulator Release 3.09.0 and to all subsequent versions, releases and modifications until otherwise indicated in new editions. Make sure you are using the correct edition for the level of software you are using.

## 1.2 What this book is about

This book is a guide to installing the Hercules Emulator and related additional products (both required and optional) under the Microsoft Windows, Linux and Mac OS X operating systems. For guidance in operating or debugging Hercules or for a general overview additional manuals are available.

Please note that some information can be found in more than one manual. This redundancy is not intended to unnecessarily expand the manuals, rather to help to find all necessary information in one place.

## 1.3 Who should read this book

This book is mainly intended for people who are responsible for installation and maintenance of the Hercules Emulator. It may also be useful if you are responsible for operating the Hercules Emulator.

## 1.4 What you need to know to understand this book

To understand this book you should be familiar with installing software under the Windows, Linux or Mac OS X operating systems. You should also have some experience in using command shells under the mentioned operating systems. Some knowledge of TCP/IP configuration in a small network is required to configure network connectivity.

Last but not least you should be familiar with IBM mainframe environments (hardware and software) and the underlying ideas and concepts as Hercules emulates IBM mainframe hardware.

## 1.5 How to use this book

This book is designed as a step by step installation guide for the Hercules Emulator and related products. You should go through the book chapter by chapter and follow all the instructions given. This should lead to an easy and fast installation without major problems.

## 1.6 Revision Notice

Hercules Release:      Version 3 Release 09 Modification 0

Publication Number:    HEIG030900

SoftCopy Name:         HerculesInstallation

Revision Number:       HEIG030900-00

Date:                  July 15, 2013

## 1.7 Readers Comments

If you like or dislike anything about this book please send an email to the address below. Feel free to comment on any errors or lack of clarity. Please limit your comments on the information in this specific book and also include the "Revision Notice" just above. Thank you for your help.

Send your comments by email to the Hercules-390 discussion group:

hercules-390@yahoogroups.com

## 1.8 Legal Advice

Hercules implements only the raw S/370, ESA/390, and z/Architecture instruction set, it does not provide any operating system facilities. This means that you need to provide an operating system or standalone program which Hercules can load from an emulated disk or tape device. You will have to write the operating system or standalone program yourself unless you possess a license from IBM to run one of their operating systems on your PC or use IBM programs and operating systems which have been placed in the public domain.

*NOTE: It is <u>YOUR</u> responsibility to comply with the terms of the license for the operating system you intend to run on the Hercules Emulator.*

## 1.9 Trademarks

The following is a list of trademark acknowledgments and copyright notices for product and company names mentioned in this book. Other product and company names in this book that are not listed below may be the trademarks or registered trademarks of their respective owners.

- IBM, System/370, ESA/390, z/Architecture, MVS, OS/390, z/OS, VM, VM/ESA, z/VM, VSE, VSE/ESA, z/VSE are trademarks or registered trademarks of International Business Machines Corporation (IBM).

- Windows XP, Windows Vista, Windows 7, Windows Server 2003, Windows Server 2008, Visual C++ Toolkit, Visual C++ Express are trademarks of Microsoft Corporation.

- Linux is a trademark owned by Linus Torvalds. The Linux Mark Institute is the exclusive licensor of the Linux trademark on behalf of its owner Linus Torvalds.

- WinPcap is copyrighted by NetGroup, Politecnico di Torino (Italy).

- Cygwin is copyrighted by Red Hat, Inc.

- Vista tn3270 is copyrighted by Tom Brennan Software.

- Pentium, XEON are trademarks or registered trademarks of Intel Corporation.

- Athlon, Opteron are trademarks or registered trademarks of Advanced Micro Devices (AMD), Inc.

- Xmit Manager is copyrighted by Neal Johnston-Ward.

- FLEX-ES is a registered trademark of Fundamental Software, Inc.

- UMX Virtual Mainframe is a registered trademark of UMX Technologies.

## 1.10 Acknowledgements

The Hercules manuals would not have been possible without the assistance of many people and I would like to thank all those who helped me. In particular I would like to thank:

- The Hercules developers for their documentation on various websites from which I derived a great deal of information.

- Roger Bowler and Fish for proof-reading the manuals.

- Loris Degoianni for allowing me to use parts of the original WinPcap documentation.

- Tom Brennan for allowing me to use parts of his Vista tn3270 documentation.

- My colleagues for working with early previews of the documentation, beginning with just a few pages.

- Mike Cairns for reviewing and editing the manuals.

- Robert Allan for providing the "Linux Installation" part.

- Lutz Mader for providing the "Mac OS X Installation" part.

If anyone feels they have been forgotten on this list please let me know.

Peter Glanzmann

# 2. Related Publications

## 2.1 Hercules Emulator – General Information

The Hercules "General Information" manual provides an overview of the ideas and concepts of the Hercules Emulator as well as documentation of the emulators functionality. It explains what Hercules does and does not and helps you decide if the software fits to your needs and if it can fulfill all your requirements.

## 2.2 Hercules Emulator – Installation Guide

The Hercules "Installation Guide" shows you how to install Hercules and all related optional and required software components under the Microsoft Windows, Linux and Apple MacIntosh OS X operating systems.

After going through the installation guide you will have a working emulator environment ready to IPL a S370, S/390 or z/Architecture mainframe operating system.

## 2.3 Hercules Emulator – User Reference Guide

The Hercules "User Reference" leads you through all aspects of the emulator's operation. It provides instruction in the operation of the Hercules Emulator with and without the Windows GUI. The usage details for all Hercules utilities are also covered in this guide.

After reading this manual you should be able to work with Hercules and the Hercules console, create virtual devices, understand backup / restore procedures and general housekeeping within the Hercules environment.

## 2.4 Hercules Emulator – Messages and Codes

The "Messages and Codes" manual provides a detailed explanation of all Hercules related messages. It is the primary source for troubleshooting and debugging when you experience problems with Hercules.

## 2.5 Hercules Emulator – Reference Summary

The Hercules "Reference Summary" booklet lists all the system parameters, device definitions, console commands, Hercules utilities etc. along with their arguments.

This booklet is intended as a quick reference guide for experienced users. Consult the Hercules "User Reference Guide" for more detailed and additional information.

# Part I: Hardware and Performance

# 3.   Hardware Prerequisites

## 3.1   PC Hardware

The following section lists the requirements for various hardware components within the context of the following categories:

- Minimal          (minimum required equipment)
- Moderate         (average older PC equipment)
- Average          (typical current PC equipment)
- Good             (fast, newer PC environment)
- Optimal          (recommended server equipment for best performance)

It is possible to run Hercules with less than the minimal recommended hardware but performance of any operating system executing under the emulator will be severely constrained. For acceptable performance you will need a configuration categorized as "average" or better. If you have access to PC hardware categorized as "optimal" the performance of installed OS's will be adequate for most practical purposes.

Most modern PC systems are delivered with adequate RAM, Processor and Disk for Hercules use. It has been stated in previous Hercules documentation that a Pentium 200 MHz with 32 MB of RAM would be sufficient for the emulator, however today such a system is no longer practical.

### 3.1.1  Processor

Hercules does not necessarily depend on the Intel Pentium architecture. It also has been built, installed and run successfully on an Alpha 21164, SPARC and on z/Architecture Linux/390 systems. One of the most extravagant implementations for testing purposes has run OS/360 under Hercules under Linux/390 under Hercules under Linux/390 under VM/ESA. This is of course an extreme example of emulation layers. This document however only describes the implementation of the Hercules Emulator on an Intel based or compatible PC system.

Hercules will benefit greatly from a fast processor, the faster the processor the better Hercules will run. If you can employ a multiprocessor or dual-core system, so much the better. The Hercules Emulator makes extensive use of multithreading to overlap I/O with CPU activity. A multiprocessor board with two slower processors will in most cases outperform a uniprocessor board with a faster processor.

- Pentium with 500 MHz or equivalent processor (minimal)
- Pentium with 1 GHz or equivalent processor (moderate)
- Pentium with 2 GHz or equivalent processor (average)
- Pentium with 3 GHz (Dual or Quad Core) or equivalent processor (good)
- 2 or more Pentium Quad Core processors with 2.66 GHz or equivalent processors (optimal)

### 3.1.2  RAM

The more RAM installed in the system the better Hercules will perform. For maximum throughput you should set your main and expanded storage sizes in the Hercules configuration file high enough to eliminate installed operating system paging operations as much as possible. The S/390 guest system storage is allocated out of the Linux or Windows host operating system storage so try to provide enough RAM to your system to eliminate Linux or Windows paging as well.

- 128 MB RAM (minimal)

- 512 MB RAM (moderate)

- 1024 MB RAM (average)

- 2048 MB RAM (good)

- 4096 MB RAM (optimal)

There is a limit to the memory that the 32-bit version of Hercules can allocate under Windows (usually around 1 GB), therefore normally 2 GB of installed RAM in the machine should be sufficient to allow for both Hercules and the host OS. If however you plan to run more than one instance of Hercules on the same machine, each instance can allocate 1 GB of RAM for its own use. In this case it makes sense to install more than 2 GB RAM. This limitation does not apply when using the 64-bit version of Hercules.

Please note that you still can allocate more RAM than is physically available on the PC to the Hercules emulator for use of the installed operating system(s). But in this case the Linux or Windows host operating system has to page out the missing physical RAM to its own swap files which will seriously degrade Hercules performance.

## 3.1.3  Disk Storage

The disk storage requirements for Hercules depend entirely on your requirements. The runtime Hercules system requires only small amount of disk space. You will need a little more space if you plan to build Hercules from source code. But still the requirements are quiet modest.

Although Hercules and the necessary software components do not require very much disk storage, you need enough hard disk space to accommodate the emulated DASD volumes for the operating system you choose to run under Hercules.

For a minimal z/OS system with CICS, IMS, DB2, WAS etc., without any user data, you need at least 15 disks of type 3390 model 3 which requires 42.5 GB hard disk space. If you extend such a system with some user data for software development etc. you may need 10 to 15 additional 3390 model 3 disks. This could lead to a total hard disk requirement of 85 GB. If you plan to do uncompressed backups of all the DASD volumes you can double this number.

The following table shows how much space is occupied for each virtual DASD volume on your PC hard disk(s) for some of the supported device types. If you make use of the compressed CKD DASD feature of Hercules these sizes will shrink dramatically, usually to about 20 to 30 percent of the original size. Space savings depend on the actual used capacity within the virtual DASD volumes.

| Model | Cylinder | Bytes/Track | Bytes/Cylinder | Bytes/Volume |
|---|---|---|---|---|
| 3380-J | 885 | 47'476 | 712'140 | 630 MB |
| 3380-E | 1'770 | 47'476 | 712'140 | 1.26GB |
| 3380-K | 2'665 | 47'476 | 712'140 | 1.89 GB |
| 3390-1 | 1'113 | 56'664 | 849'960 | 946 MB |
| 3390-2 | 2'226 | 56'664 | 849'960 | 1.89GB |
| 3390-3 | 3'339 | 56'664 | 849'960 | 2.83 GB |
| 3390-9 | 10'017 | 56'664 | 849'960 | 8.51 GB |
| 3390-27 | 32'760 | 56'664 | 849'960 | 27.84 GB |
| 3390-54 | 65'520 | 56'664 | 849'960 | 55.68 GB |
| 9345-1 | 1'440 | 46'456 | 849'960 | 1.0 GB |
| 9345-2 | 2'156 | 46'456 | 849'960 | 1.5 GB |

**Table 1: DASD Device Capacity**

# 4.  Performance

As described previously the performance of the Hercules Emulator depends heavily on the underlying PC hardware. It is therefore not possible to give exact performance specifications of any particular operating system running under Hercules. Some practical values from user experiences across several machines presented here provide some guidelines though.

The performance of Hercules is measured in two values, MIPS and I/O rate. Both of these values are presented on the Hercules console and are refreshed every second independently of the PANRATE control statement. See Hercules User Reference for details on the PANRATE statement.

## 4.1  MIPS

MIPS is an abbreviation of "Million Instructions Per Second" and presents a measure of the number of instructions the CPU is executing in one second. MIPS is a measure of a computer's processor speed. However this measure is useful only among processors with the same instruction set as different instruction sets take different numbers of instructions to do the same job. Many of the reported MIPS values represent 'Peak' execution rates on artificial instruction sequences with few branches, whereas realistic workloads consist of a mix of instructions some of which take longer to execute than others.

The performance of the memory hierarchy greatly affects processor performance, an issue also not considered in simple MIPS comparisons. In an attempt to address these issues researchers have created standardized tests such as "SpecInt" to measure the real effective performance in commonly used applications. The use of raw MIPS as a measure of overall system performance has fallen into disuse. MIPS is sometimes pejoratively referred to as "Meaningless Indicator of Processor Speed" or "Meaningless Information Provided by Salespeople".

The Hercules console reports the MIPS rate for the emulated S/370, ESA/390 or z/Architecture instructions, not the underlying executed instructions of the hardware. As implied previously the MIPS rate can vary significantly depending on the executed instruction and whether the instructions can be processed entirely in cache, as can happen in a tight loop.

It is difficult to determine how the speed of the Hercules emulation corresponds to a real mainframe. This is due to difficulties in comparing real mainframe hardware to PCs (or servers) as well as the actual performance measurement itself. Hercules shows its processing speed in MIPS. Compared to the earlier IBM System/360 and System/370 hardware, it is safe to say that Hercules will outperform them when it is running on moderately powerful hardware, whereas newer IBM System/390 and z/Series hardware still is much faster than the emulation.

On a Celeron 300 you should see an execution speed of 1 to 2 MIPS, which is enough to run OS/360 (MFT or MVT) or MVS 3.8 with a response time better than that of a 3033 from the 1970's. It's also fast enough to run VSE/ESA with an acceptable response time. On a more recent system with a 2 GHz Pentium processor, you may see the system peak at around 30 MIPS which is enough to run Linux/390 or z/OS with a light workload.

Performance on server class machines is now fairly respectable. For example, on a dual-core Intel Xeon with "Hyperthreading" (4 CPUs) running at 3.46 GHz, you might expect to see a sustained MIPS rate of 40 to 60 MIPS. A dual-processor quad-core Mac Pro (8 cores, 3 GHz) will sustain over 150 MIPS. For anyone who is prepared to spend a considerable amount of money on their Hercules system, there are reports that a sustained 300+ MIPS rate has been achieved on an Intel Core i7 processor running at 3.75 GHz using all four cores plus "Hyperthreading" (8 CPUs).

 The speed however depends greatly on the executed instructions. Instructions that are very expensive to emulate can even on fast systems still be around 1 - 2 MIPS, see the tables further below for details.

## 4.2  I/O Rate

The second value that gives us performance data is the I/O rate. This is the average number of SIOs (Start I/O per second) occurring to active devices, including DASD, TAPE, CTC (channel-to channel adaptor), local SNA and non-SNA devices etc.

This value also varies depending on the underlying hardware. A RAID system will easily outperform a non-RAID system. Many PC systems today can be ordered with a RAID-0 (Striping) or RAID-1 (Mirroring) adapter. While disk mirroring (RAID-1) gives fault tolerance it does not improve performance. However Data striping (RAID-0) spreads blocks of each file across multiple disks and can nearly double the performance of a single disk.

On a recent PC system as described above, incorporating two Serial-ATA (SATA) disk drives connected to a RAID adapter card with activated RAID-0 has shown peak I/O rates of more than 3250 SIOs per second. The sustained I/O rate delivered from such a system was more than 1500 SIOs per second.

## 4.3  Hercules Performance Measurements

The following sections show some performance index tables under various system configurations. Individual measured performance may vary from the figures shown as performance depends heavily on the hardware used and the Hercules release level.

All tests have been performed with the following measurement tools:

- IMON (Rate CPU Instruction Speed)
- CPU Instruction Timing Tool (INSTRATE program)

Tests showing the performance of Hercules itself include measurements of the Hercules Emulator software and the difference between working with CKD or CCKD (compressed CKD) DASD emulation. Other tests, like the host CPU on which Hercules runs or the disk types of the host system, show the influence of the hardware used. The last test ("Hercules Overall Performance") represents the practical performance growth of the Hercules Emulator as both hardware and software have improved.

### 4.3.1  Hercules Emulator Performance

The following diagram shows the software related performance index of the Hercules Emulator software. The graphic shows the performance improvements that have been made over recent releases. The base version (index = 100%) for all measurements was Hercules V3.00.

All these tests were performed using the same hardware configuration. The results therefore demonstrate the influence of the improvements in the performance of the Hercules Emulator software only.
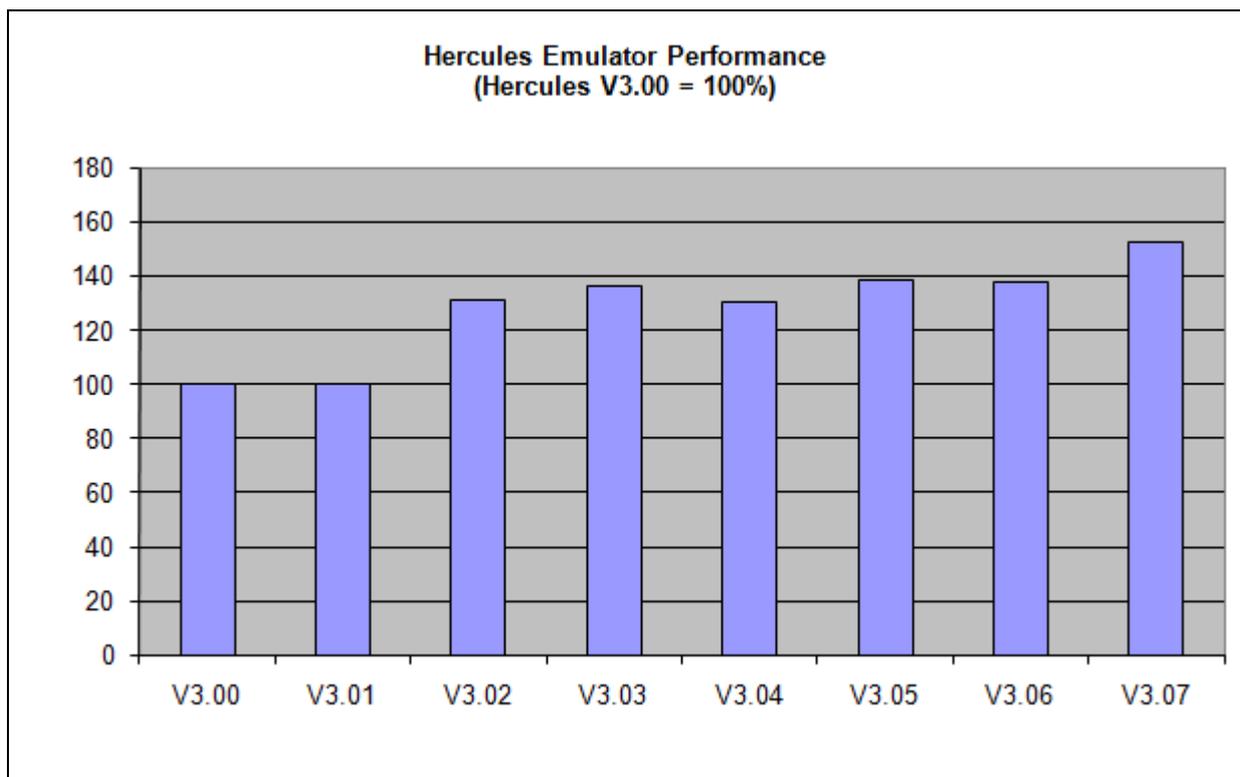
**Figure 1: Hercules Emulator Performance**

Although there have been massive performance improvements in the Hercules Emulator software in recent releases, the current rate of improvement cannot be expected to continue. On one hand, the maximum performance improvements available via software improvements will be reached, on the other hand the developers of Hercules have architectural frameworks that impose certain limitations, as explained below.

Firstly the Hercules software should be portable and is therefore written entirely in "C". The use of assembler to give additional performance is not exploited. Secondly the developers try to emulate real hardware as exactly as possible in accordance with IBM's "Principles of Operations" documentation. This leads to code that may be not optimal for the underlying hardware, but provides perfect emulation of real mainframe hardware, avoiding problems when running mainframe operating systems.

The performance of the Hercules Emulator software depends heavily on the set of executed (mainframe) instructions as detailed in below ("Emulated Instruction Performance").

## 4.3.2  Emulated CKD / CCKD DASD Performance

The type of the emulated DASD devices, either CKD or CCKD, has direct influence on performance. If using CKD devices there is more data to be read from the disk and transferred to memory, but the data is directly usable and no further processing needs to be done. Using CCKD (compressed CKD) devices the amount of data to be read from the disk and transferred to memory is greatly reduced but once in memory the data has to be uncompressed before it can be processed.

The performance characteristics of DASD types are irrelevant if host disk space limitations impose the choice of using compressed CKD only.

On a fast machine with fast disks CKD is the best choice. On a fast CPU with slow disks however, CCKD can give better overall performance. A fast processor can uncompress the data in less time than that saved by transferring the data uncompressed. In the less common case of a slow CPU with fast disks

CKD DASD can perform better as the uncompressed data is transferred faster than the processor could uncompress it. In the case of a slow machine with slow disks and possibly space constraints it is recommended to work with CCKD DASD.

Another point to consider is reliability. Although Hercules itself is very stable, occasional machine crashes such as the Windows "Blue Screen of Death" can occur. The CKD DASD emulation in these cases is very stable; it is usually possible to just restart the machine with no problems. However when working with CCKD DASD emulation, after a crash the CCKD routines have to perform a recovery during the restart of Hercules. This recovery takes from seconds to several minutes depending on what happened and the number of defined DASD devices. Although this recovery is normally successful cases have been reported where the recovery failed. In this case the only option is to restore DASD images from previously saved backups.

The following figure shows a performance comparison of compressed DASD (CCKD) versus CKD devices. The IPL time with CKD devices was 75 seconds. The IPL time with CCKD devices is only slightly higher (about 80 seconds).
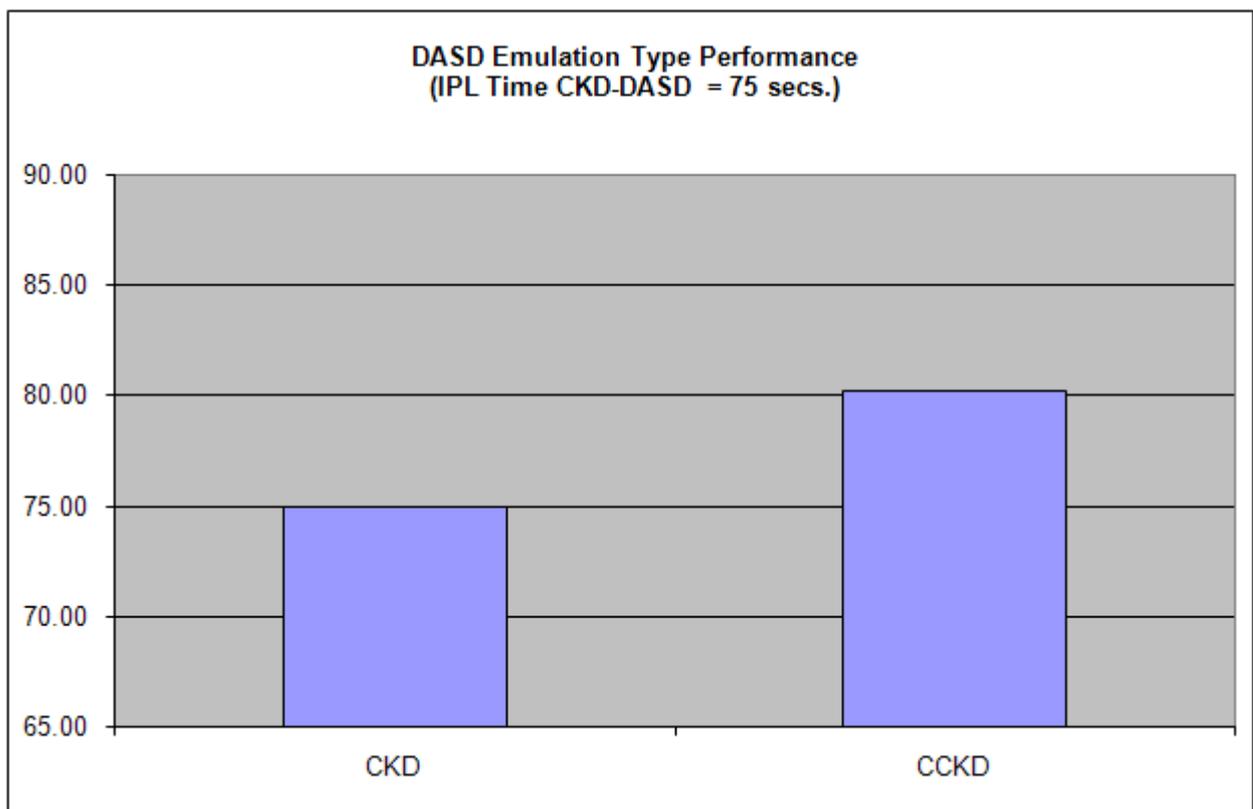


 **Figure 2: DASD Emulation Type Performance**

Both tests were performed on the same machine (2 x Opteron 280, 2.4 GHz with a Software RAID-0 hard disk configuration). The measured performance degradation using CCKD emulation over CKD is only approximately 7.0 percent.

## 4.3.3 Host CPU Performance

The following diagram shows the CPU related performance results for Hercules. The base CPU at index = 100% is an Intel Pentium 3 with 500 MHz clock speed.

All these tests were performed with the same Hercules release (V3.05) and therefore show only the influence on performance of the real CPU and I/O devices.
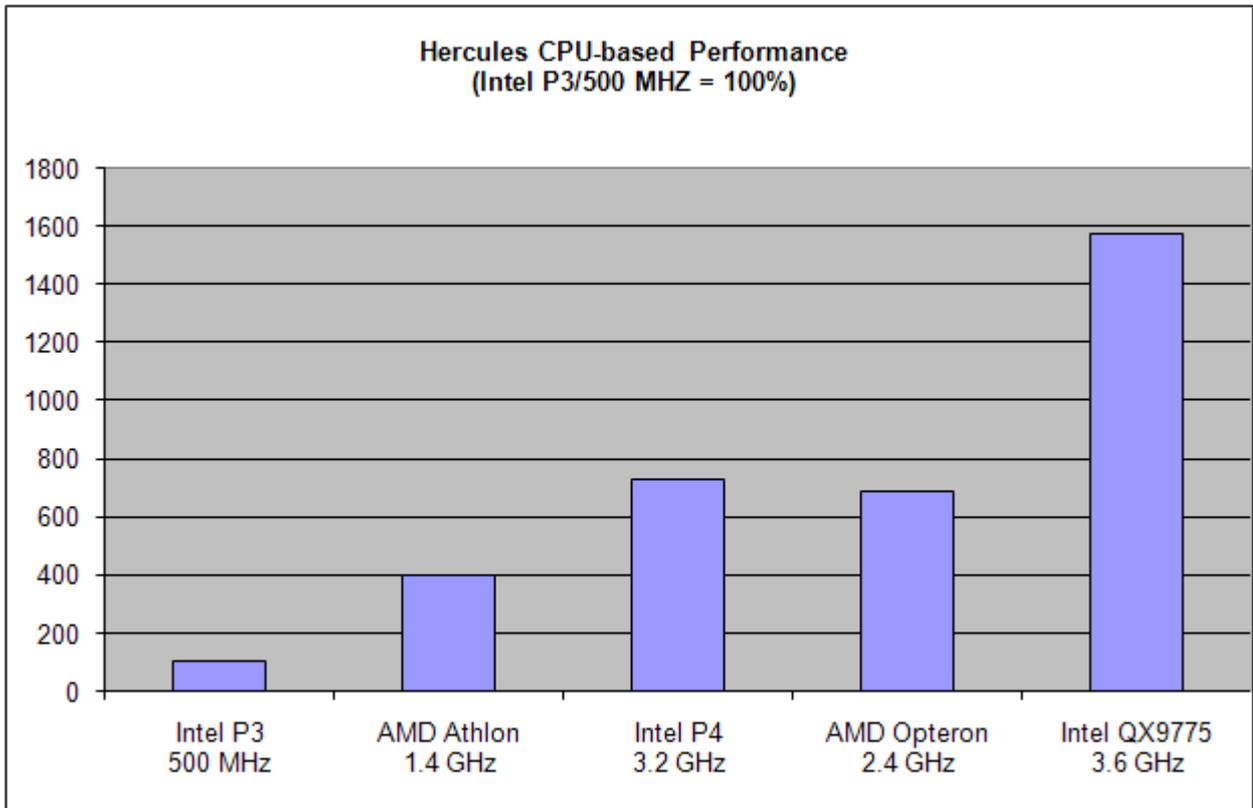


**Figure 3: Hercules CPU-based Performance**

The figures shown above are influenced by the I/O configuration used. As mainframe operating systems running under Hercules are often heavily I/O based, bare CPU speed is not the only relevant performance factor. An additional performance boost is achieved using RAID-0 striping and / or using several physical disks in the host system under which the emulator is running, as discussed in the following section.

## 4.3.4  Host Disk Performance

The following table shows the different disk and controller configurations that have been used for testing the disk transfer rates. The comparison of the reached transfer rates is visualized in the figure following the configurations table.

| Test Scenarios | Scenario 1 | Scenario 2 | Scenario 3 | Scenario 4 |
|---|---|---|---|---|
| Disk Type | Western Digital WD1500ADFD | Western Digital WD1500ADFD | Hitachi Ultrastar 15k300 | Hitachi Ultrastar 15k300 |
| Disk Controller | Nvidia nForce4 SATA | Nvidia nForce4 SATA | Adaptec RAID 3405 | Adaptec RAID 3405 |
| RAID Configuration | none | RAID-0 | none | RAID-0 |

**Table 2: Disk Configurations**

These configurations reached the transfer rates as shown in the following figure.
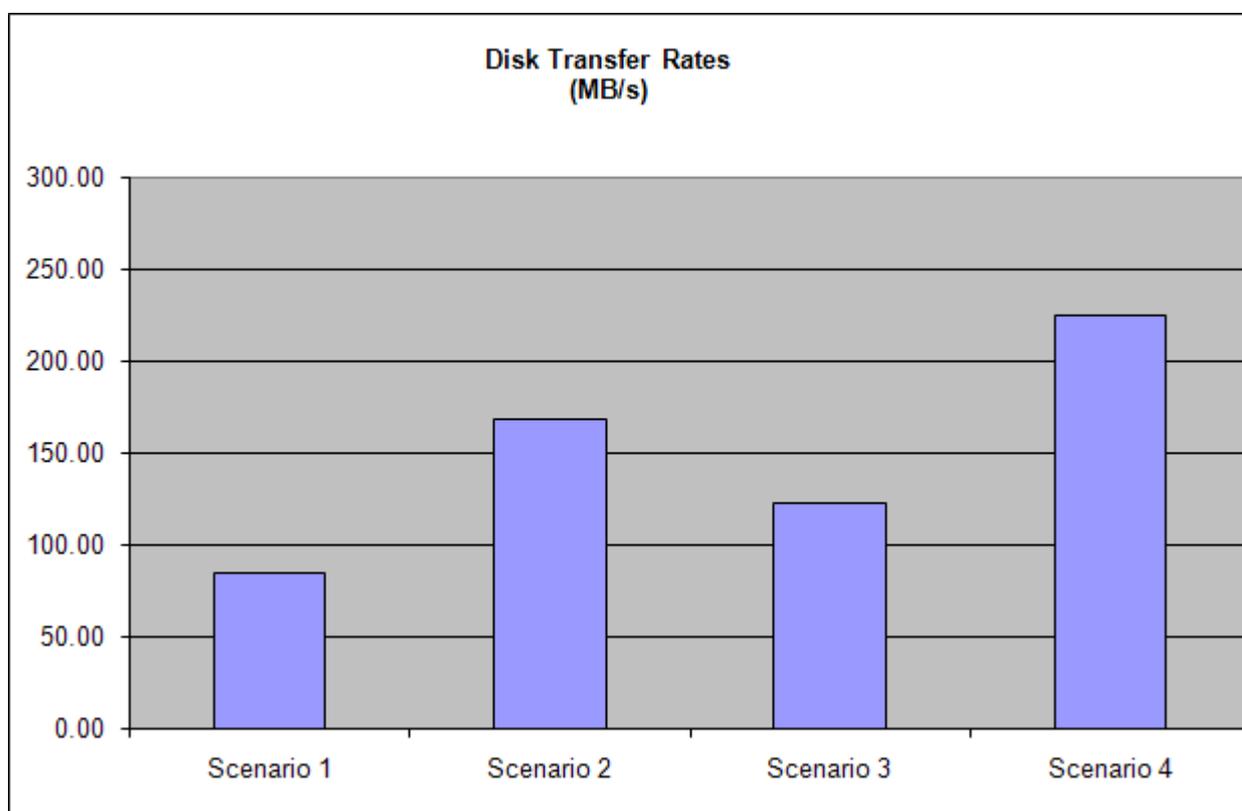


**Figure 4: Disk Transfer Rates**

While these measurements only show the "raw speed" of the disks without direct relationship to Hercules, the next diagram shows the impact on the IPL time depending on the used disk configuration. Both tests have been run using the same hardware (2 x AMD Opteron 280, 2.4 GHz), the only change being the type of disk employed.

The first test was made with a non-RAID disk configuration (Scenario 1, see above) which reached a transfer rate of 85 MB/s. The second IPL was done with a RAID-0 disk configuration (Scenario 2, see above) with a transfer rate of 169 MB/s. The IPL with the RAID-0 disk configuration needed only 75 seconds compared to 81 seconds with the non-RAID setup.
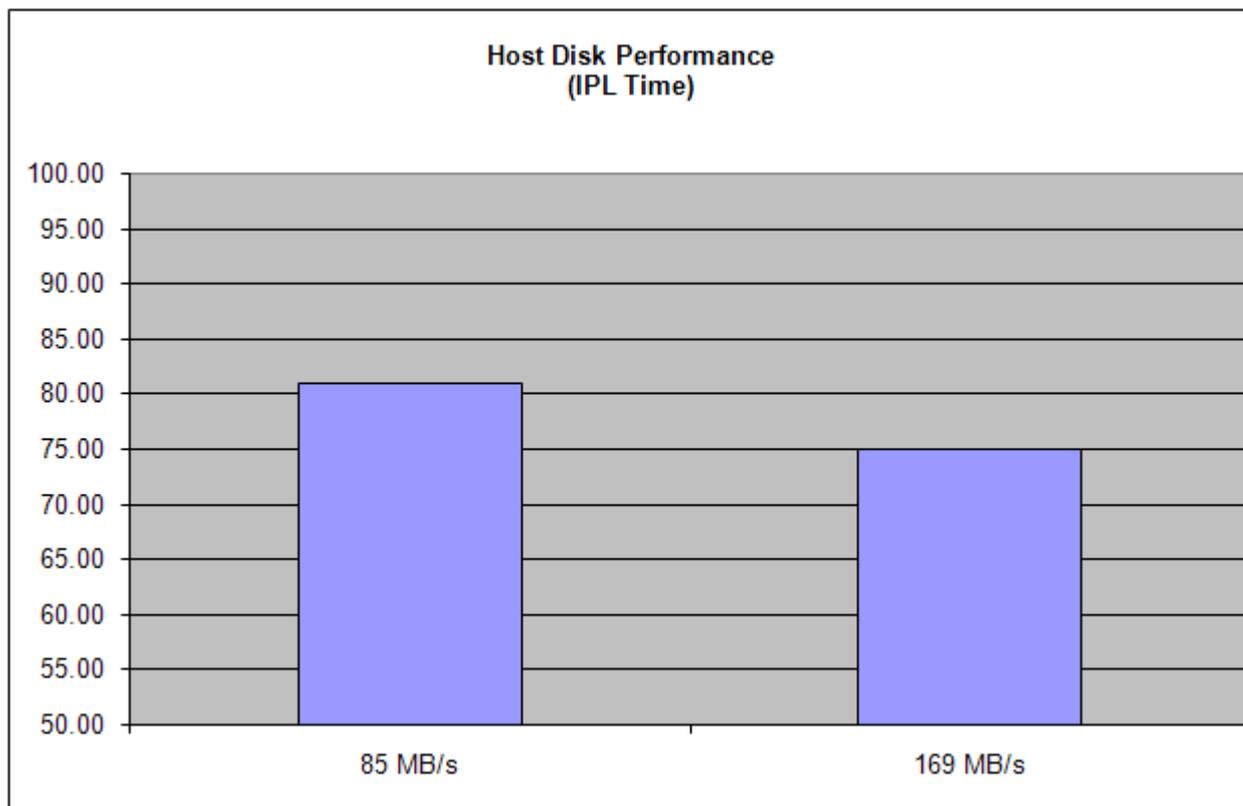


**Figure 5: Host Disk Performance (IPL in seconds)**

These tests demonstrate that fast disks and RAID-0 configurations provide a performance boost from which Hercules will benefit.

## 4.3.5 Emulated Instruction Performance

As previously stated, Hercules Emulator performance depends heavily on the mainframe instructions being executed. While some S/3xx instructions are relatively easy to implement and require only a few "C" instructions in the emulator, others are quite complicated and require a lot of instructions to emulate the real hardware. Due to this fact the performance of each individual S/3xx instruction can vary greatly. There is a variance of more than 450 times between the fastest and slowest emulated instructions.

It is known though, that in mainframe operating systems, a small subset of the available instructions are employed to do most processing. In Hercules, if most of these instructions are emulated slowly, the overall performance of the emulator will be poor. Alternatively, if the most commonly used instructions are emulated quickly the performance of the emulator will benefit greatly.

An analysis of mainframe operating systems has shown that only 5 different instructions are responsible for up to 50% of the executed code. Expanding this analysis to 10-15 instructions will cover up to 75% of executed code and 15-20 instructions will cover more than 80% of all mainframe code.

The following tables list some different test scenarios and the performance ratios of individual instructions and real work tests under the Hercules Emulator. The used hardware for each scenario is listed in the following table. All test jobs have been run three times under each scenario. The average MIPS rate of these three runs has been taken into the table.

These tests are not meant to provide a definitive MIPS rating (your own measurements may be some-what different), but serve as a repeatable test for comparing the relative speeds of different hardware.

| Test Scenarios | Scenario 1 | Scenario 2 | Scenario 3 |
|---|---|---|---|
| PC Model (month/year) | 08/2003 | 09/2006 | 06/2008 |
| CPU Type | Intel Pentium P4 | AMD Opteron 280 | Intel QX9775 EE |
| CPU Clock Rate | 3.2 GHz | 2.4 GHz | 3.6 GHz |
| Number of physical CPUs | 1 | 2 | 2 |
| Number of CPU Cores | 2 | 4 | 8 |
| RAM | 2 GB | 8 GB | 8 GB |
| Operating System | Windows XP Pro | Windows XP Pro x64 | Windows XP Pro x64 |
| Hercules Release | V 3.05 | V 3.05 | V 3.05 |

**Table 3: Performance Test Scenarios**

The first table of performance results shows the reachable MIPS rates of various single instructions. The test is based on a modified INSTRATE program, originally written by Gary Brabiner.

| Executed Instruction | Scenario 1 | Scenario 2 | Scenario 3 |
|---|---|---|---|
| BCT   Rx,LOOP | 99.55 MIPS | 84.06 MIPS | 82.14 MIPS |
| BCTR  RX,RLOOP | 109.10 MIPS | 93.86 MIPS | 137.74 MIPS |
| NOP   R0 | 154.84 MIPS | 144.52 MIPS | 431.59 MIPS |
| LR    R1,R0 | 66.33 MIPS | 58.07 MIPS | 101.75 MIPS |
| LTR   R1,R0 | 64.76 MIPS | 58.12 MIPS | 102.03 MIPS |
| L     R1,0 | 10.09 MIPS | 12.43 MIPS | 18.97 MIPS |
| L     R1,DATA | 39.00 MIPS | 33.87 MIPS | 42.65 MIPS |
| L     R1,DATA+1 | 37.95 MIPS | 33.44 MIPS | 41.11 MIPS |

| Executed Instruction | Scenario 1 | Scenario 2 | Scenario 3 |
|---|---|---|---|
| LH     R1,DATA | 37.74 MIPS | 33.93 MIPS | 49.96 MIPS |
| ICM    R1,15,DATA | 35.50 MIPS | 33.85 MIPS | 39.89 MIPS |
| ICM    R1,1,DATA | 20.28 MIPS | 20.96 MIPS | 29.53 MIPS |
| IC     R1,DATA | 37.59 MIPS | 33.77 MIPS | 51.27 MIPS |
| LD     F0,DATA | 13.95 MIPS | 25.23 MIPS | 51.50 MIPS |
| LM     8,6,SAVEREGS+(8*4) | 18.78 MIPS | 17.98 MIPS | 19.34 MIPS |
| STM    1,14,DATA | 24.01 MIPS | 19.21 MIPS | 20.12 MIPS |
| ST     R1,DATA | 37.77 MIPS | 32.91 MIPS | 44.73 MIPS |
| STH    R1,DATA | 36.84 MIPS | 34.04 MIPS | 44.61 MIPS |
| STCM   R1,15,DATA | 35.87 MIPS | 28.04 MIPS | 40.87 MIPS |
| STCM   R1,1,DATA | 22.51 MIPS | 19.59 MIPS | 36.65 MIPS |
| STCM   R1,8,DATA | 23.05 MIPS | 20.52 MIPS | 42.67 MIPS |
| STC    R1,DATA | 39.97 MIPS | 34.02 MIPS | 46.95 MIPS |
| MVI    DATA,CHAR | 39.77 MIPS | 35.70 MIPS | 56.72 MIPS |
| MVC    DATA(8),DATA | 19.72 MIPS | 15.57 MIPS | 28.47 MIPS |
| MVC    DATA(32),DATA | 13.12 MIPS | 11.11 MIPS | 15.72 MIPS |
| MVC    DATA(32),DATAB | 2.25 MIPS | 6.05 MIPS | 30.65 MIPS |
| MVC    DATA(32),0 | 1.94 MIPS | 4.74 MIPS | 14.68 MIPS |
| MVC    DATA(256),DATA | 3.56 MIPS | 3.59 MIPS | 4.88 MIPS |
| MVC    DATA(256),DATAB | 0.31 MIPS | 1.16 MIPS | 15.45 MIPS |
| XC     DATA(4),DATA | 17.58 MIPS | 19.12 MIPS | 37.05 MIPS |
| XC     DATA(4),DATAB | 22.61 MIPS | 22.12 MIPS | 26.97 MIPS |
| CLI    DATA,CHAR | 39.73 MIPS | 32.00 MIPS | 46.08 MIPS |
| CLC    DATA(8),DATA | 26.42 MIPS | 23.25 MIPS | 36.16 MIPS |
| CLC    DATA(32),DATA | 22.15 MIPS | 17.85 MIPS | 35.87 MIPS |
| CLC    DATA(32),DATAB | 22.53 MIPS | 17.80 MIPS | 36.51 MIPS |
| CLC    DATA(32),0 | 10.01 MIPS | 10.43 MIPS | 15.78 MIPS |

| Executed Instruction | Scenario 1 | Scenario 2 | Scenario 3 |
|---|---|---|---|
| CLC    DATA(256),DATA | 8.84 MIPS | 7.89 MIPS | 19.67 MIPS |
| CLC    DATA(256),DATAB | 8.88 MIPS | 7.90 MIPS | 19.37 MIPS |
| AR     R1,R0 | 49.39 MIPS | 37.65 MIPS | 78.44 MIPS |
| ALR    R1,R0 | 59.98 MIPS | 56.69 MIPS | 78.85 MIPS |
| A      R1,F1 | 36.09 MIPS | 32.92 MIPS | 41.32 MIPS |
| AH     R1,H1 | 35.57 MIPS | 30.76 MIPS | 41.28 MIPS |
| LA     R1,1(,R1) | 55.35 MIPS | 45.18 MIPS | 71.40 MIPS |
| LA     R1,1(R1) | 55.46 MIPS | 48.94 MIPS | 63.86 MIPS |
| AP     PL4,PL4 | 1.67 MIPS | 1.69 MIPS | 2.61 MIPS |
| AP     PL4,PL4B | 1.87 MIPS | 1.80 MIPS | 3.02 MIPS |
| AP     PL16,PL16 | 1.46 MIPS | 1.52 MIPS | 2.27 MIPS |
| AP     PL16,PL16B | 1.64 MIPS | 1.69 MIPS | 3.02 MIPS |
| CVD    R1,DATA | 2.82 MIPS | 3.09 MIPS | 21.44 MIPS |
| CVD    R15,DATAB | 6.60 MIPS | 7.12 MIPS | 26.17 MIPS |
| AER    F0,F4 | 32.26 MIPS | 30.40 MIPS | 40.82 MIPS |
| ADR    F0,F4 | 25.78 MIPS | 25.75 MIPS | 45.64 MIPS |
| AXR    F0,F4 | 16.69 MIPS | 17.78 MIPS | 43.74 MIPS |
| Move  256 Bytes by MVC | 0.31 MIPS | 1.16 MIPS | 15.84 MIPS |
| Move  256 Bytes by MVCL | 3.25 MIPS | 4.84 MIPS | 11.48 MIPS |
| Clear 256 Bytes by MVC | 0.93 MIPS | 1.32 MIPS | 15.23 MIPS |
| Clear 256 Bytes by MVCL | 3.63 MIPS | 6.11 MIPS | 16.54 MIPS |

**Table 4: Emulated Instruction Performance**

The next table shows some performance data from the "CPU Speed Rating Monitor" of IMON (Interactive Monitor IMON/370 and IMON for OS/390 and z/OS, see http://www.prycroft6.com.au).

The eight tests of IMON run the following instructions:

- RR instructions (fast):
  ALR, AR, BCTR, LCR, LR, NR, OR, SLR, SR and XR.

- RR instructions (slow):
  ALR, AR, BCTR, DR, LR, MR, NR, OR, SR and XR.

- RX instructions (fast):
  A, AL, BCT, IC, L, N, O, S, SL and X.

- RX instructions (slow):
  A, BCT, CVB, CVD, D, L, M, O, S and X.

- SS instructions (fast):
  AP, CLC, ED, MVC, NC, OC, SP, TR, UNPK, XC and ZAP.

- SS instructions (slow):
  CLC, DP, ED, MP, MVC, NC, OC, TR, UNPK and XC.

- Floating Point instructions (E-format, 32-bits wide):
  AE, AER, CE, DE, DER, HER, LCER, LE, ME, MER, SE and SER.

- Floating Point instructions (D-format, 64-bits wide):
  AD, ADR, CD, DD, DDR, HDR, LCDR, LD, MD, MDR, SD and SDR.

For the RR instructions tests, the operation codes listed form 10% of the instruction mix. The main slowing factors of test 2 compared to test 1 are the presence of the Divide instruction (DR) and the Multiply instruction (MR). Something similar holds true when looking at the differences between tests 3 and test 4. In tests 5 and 6 again, Divide and Multiply are used to slow test 6 as well as increasing the length of data processed by the MVC, ED, UNPK, CLC and TR instructions (one byte in test 5, maximum length in test 6).

According to IMONs help text, the "official" MIPS rating of a CPU model is usually about 20% faster than the results of test 2 (RR Slow).

| Executed Instruction | Scenario 1 | Scenario 2 | Scenario 3 |
|---|---|---|---|
| IMON RR (Fast) | 41.98 MIPS | 67.90 MIPS | 165.78 MIPS |
| IMON RR (Slow) | 36.80 MIPS | 49.16 MIPS | 138.11 MIPS |
| IMON RX (Fast) | 37.17 MIPS | 40.87 MIPS | 64.85 MIPS |
| IMON RX (Slow) | 11.20 MIPS | 13.11 MIPS | 40.21 MIPS |
| IMON SS (Fast) | 6.65 MIPS | 7.03 MIPS | 12.52 MIPS |
| IMON SS (Slow) | 0.90 MIPS | 0.95 MIPS | 3.95 MIPS |
| IMON FP (Extended) | 22.81 MIPS | 20.73 MIPS | 41.46 MIPS |
| IMON FP (Double) | 3.35 MIPS | 4.93 MIPS | 21.60 MIPS |

**Table 5: IMON Instruction Performance Data**

The last table shows the results of tests consisting of a tight loop. These tests show the influence on raw processor speed (only one job running) as well as the influence of having a multi-processor system and running several jobs in parallel.

The last test shows the time used for an IPL of a mainframe operating system from the actual IPL com-
mand until logon to the system is possible. The IPL only contains the mainframe operating system itself,
without any optional subsystems.

| Executed Instruction | Scenario 1 | Scenario 2 | Scenario 3 |
|---|---|---|---|
| Loop 1 (1 Job) | 116.04 MIPS | 105.92 MIPS | 473.27 MIPS |
| Loop 1 (2 Jobs) | 61.30 MIPS | 210.92 MIPS | 654.23 MIPS |
| Loop 1 (4 Jobs) | --- | 420.39 MIPS | 1261.85 MIPS |
| Loop 1 (8 Jobs) | --- | --- | 1676.32 MIPS |
| IPL time | 03:10 (mm:ss) | 01:15 (mm:ss) | 00:57 (mm:ss) |

**Table 6: Real Work Performance**

These tests show that, although the raw instruction speed of the four processor machine of scenario 2
compared to the two processor machine of scenario 1 is somewhat slower as shown in table 2 above, the
system greatly benefits from the four processors when doing real work, because it can handle more tasks
in parallel.

# Part II: Windows Installation

# 5. Software Prerequisites

## 5.1 Operating System

Hercules is an open source software implementation of the mainframe System/370, ESA/390 and z/Architecture hardware. Hercules itself is not an operating system nor does it emulate a mainframe operating system. The Hercules Emulator runs under Linux on several hardware platforms including the Intel Pentium PC, under various flavors of Microsoft Windows and under MAC OS X. From the point of view of the underlying operating system the Hercules Emulator is just an application program.

Part II of this guide focuses solely on the installation of Hercules under Microsoft Windows. Details of other host operating systems are covered in Part III (Linux) and Part IV (Mac OS X) in this book. The installation of any hosted operating system and software utilities is beyond the scope of this book.

### 5.1.1 Windows Versions

The Hercules Emulator runs under Windows XP, Windows Vista, Windows 7, Windows Server 2003 and Windows Server 2008. It is generally a good idea to have a current operating system maintained with the latest fixes.

Some users experienced problems with TCP/IP functionality when running under Windows XP with Service Pack 2 while other users did not encounter these issues. The presenting problem with this configuration is: It is possible to connect from the Host operating system to the Hercules machine and vice versa, however it is not possible to get a connection to another workstation on the LAN or the Internet. Most of these issues are related to configuration problems with the Windows XP firewall, introduced with Service Pack 2. Creating correct rules in either the Windows internal firewall or an alternative firewall product solves these problems.

### 5.1.2 Stability

As Hercules appears to Windows as just another application program, it is possible to run other applications simultaneously with Hercules, however this is not recommended. While the Hercules Emulator is extremely stable software it is impossible to guarantee that other applications will not degrade Hercules performance or even crash the Windows system. The less software you have installed on your system the more stable your emulated mainframe will be.

Intensive tests have proven the stability of Hercules, one of the test suites is repeated with every new Hercules release. The tests run a mixed online and batch workload continuously and at high volumes. The base system consists of a MVS 3.8J operating system with additional components. During the test the system is loaded with between four and eight self submitting (therefore continuous) batch jobs executing in parallel. These jobs consist of a mix of large sorts, assemblies, compilation and link jobs.

Additionally there are some TSO sessions active (up to ten sessions) on which a simulated user presses enter once a second in a performance monitor application.

This workload keeps the system (emulated mainframe as well as Hercules and the base operating system) busy at nearly 100% CPU and a produces a continuous I/O rate of more than 750 SIOs. In this state the machine runs 7x24 hours during several days. One of these tests was stopped after more than 12 weeks with the system still running perfectly.

### 5.1.3 Installed Software on a Hercules System

It was recommended earlier to have only a minimum of software installed on a Hercules host machine. Additional to the software for the Hercules Emulator presented in this guide, it is recommended to have the following software installed:

- Windows XP, Windows Vista, Windows 7, Windows Server 2003 or Windows Server 2008 with latest service packs and hot fixes

- Software firewall, especially if there is no hardware firewall in the LAN

- Antivirus software with on demand and on access checks active


Depending on your requirements other commonly used utilities include:

- Network sniffer

- Performance monitor / Task monitor

- FTP program


In general though the less software is installed the better the emulated mainframe will run.

## 5.2 Drivers

This Chapter describes the device drivers that are necessary specifically for the Hercules Emulator. A device driver is a routine or a set of routines that implement the device-specific aspects of generic I/O operations. Generally the Hercules Emulator does not need any special device drivers other than those provided by the operating system. An exception is WinPcap, which is required under the Microsoft Windows operating systems to enhance networking capabilities of the native operating system.

### 5.2.1 WinPcap Packet Capture Driver

WinPcap is an architecture for packet capture and network analysis for the Win32 platform developed at Politecnico di Torino in Italy. The packet filter is a device driver that adds to Windows the ability to capture and send raw data from a network card with the possibility to filter and store the captured packets in a buffer.

WinPcap includes an API that can be used to directly access the functions of the packet driver offering a programming interface independent from the Windows operating system. It also exports a set of high level capture primitives that are compatible with libpcap, the well known Unix capture library. These functions capture packets in a way independent from the underlying network hardware and operating system.

WinPcap is a free, public system and is released under a BSD-style license. It can be downloaded from www.winpcap.org where the necessary documentation can also be found.

## 5.3 Runtime Environments

This chapter describes special runtime environments which act as a base for running the Hercules Emulator. Under a Linux-like operating system no additional runtime environment is needed. Under Microsoft Windows however, the necessary Linux POSIX threads support may be emulated as explained below.

### 5.3.1  Cygwin

Until release 3.02.0 the Hercules Emulator was designed to run only under a Linux system with POSIX threads (pthread) support. To be able to run Hercules releases below 3.03.0 under Windows it is necessary to install a runtime environment to provide this Linux compatible layer. This is where Cygwin steps in.

Cygwin is a Linux-like environment for Microsoft Windows. It consists of several DLLs which act as an emulation layer providing substantial POSIX system call functionality and a collection of common Linux tools. The Cygwin environment works with all x86 versions of Windows since Windows 95.

Cygwin is available from www.cygwin.com. The Cygwin development began 1995 at Cygnus Solutions which is now owned by RedHat Software.

Cygwin is mentioned in all current Hercules manuals for support purposes only. Since release 3.02.0 of Hercules it is recommended that Windows users install the Microsoft Visual C native binaries or Microsoft Installer (MSI) package.

---

Beginning with release 3.03.0 the Hercules Emulator no longer requires Cygwin in order to run under Windows.  It is highly recommended that Windows users of Hercules begin using this new MSVC Win-32 version instead of the Cygwin versions.

Beginning with release 3.08.0 the Hercules Emulator no longer supports the Cygwin environment.

---

## 5.4  Hercules Emulator

The Hercules Emulator consists of the following mandatory and optional components:

- Hercules Emulator (mandatory)
- Hercules Windows GUI (optional)
- CTCI-W32 (optional)
- Additional utilities (optional)

### 5.4.1  Hercules

The Hercules executables are the heart of the emulator and a mandatory component. This is the software implementation of the System/370, ESA/390 and z/Architecture mainframe hardware and processor machine code instruction set.

Hercules runs as a DOS program and comes with a semi-graphical display in a DOS window (the Hercules Hardware Console - HMC) consisting of two screens, switched between using the ESC key. When the Hercules HTTP server is running then Hercules can also be operated via a web browser interface.

The following figure shows the initial display, the Hercules console window:



**Figure 6: Hercules Hardware Console - Console window**

The next figure shows the Hercules device and status display accessed with the ESC key:



**Figure 7: Hercules Hardware Console - Device and status display**

The last figure shows the Hercules web browser interface which can be accessed when the Hercules HTTP server is configured and running:
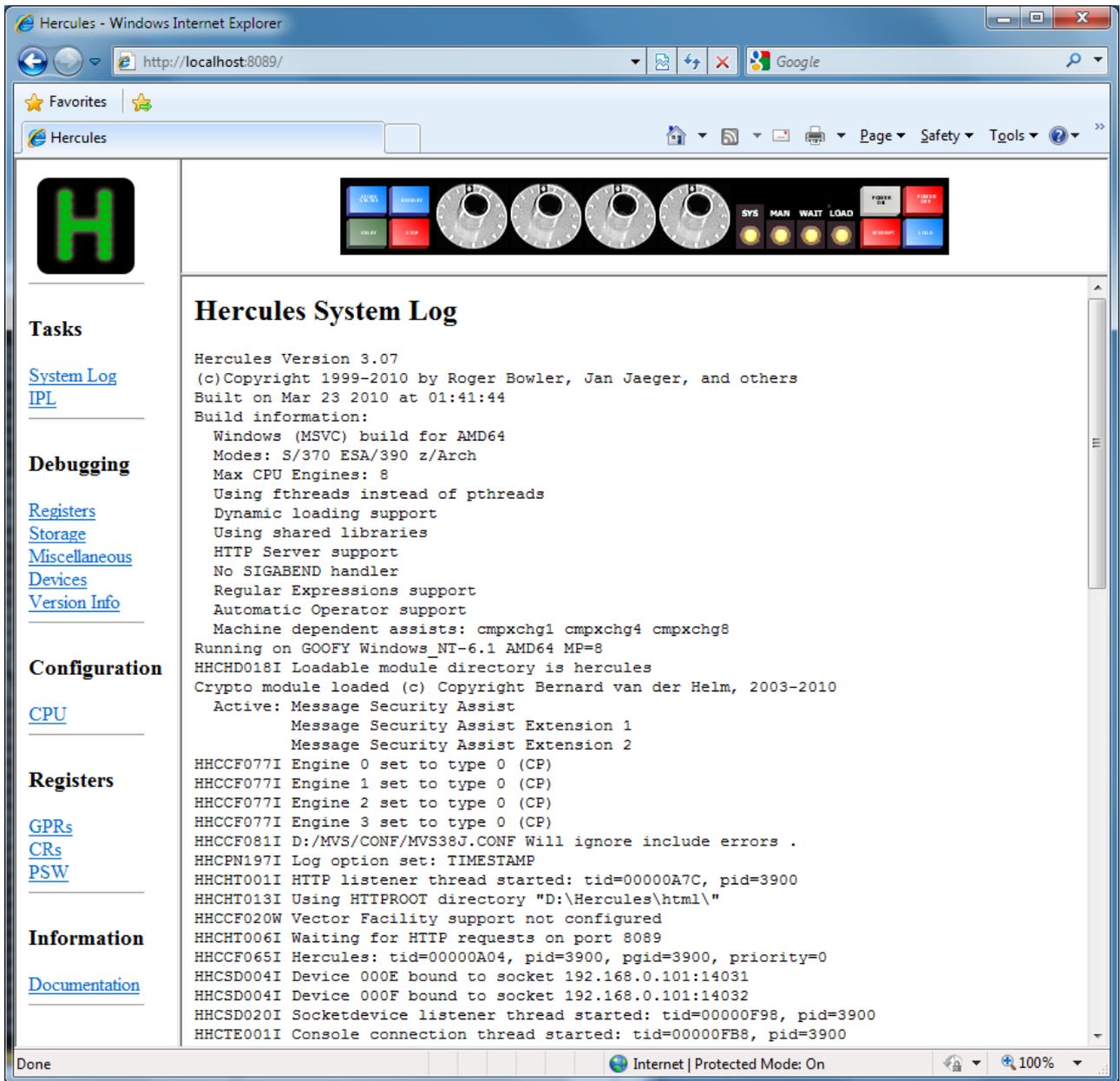


**Figure 8: Hercules web browser interface**

## 5.4.2 Hercules Windows GUI

The Hercules Windows GUI (WinGUI) provides an optional graphical user interface to the Hercules Emulator replacing the native DOS console window of Hercules. The Windows GUI program interfaces with Hercules Emulator directly. It provides an easier way to work with the Hercules Emulator including interfaces to create / change the Hercules configuration files and the handling of log files.

The following figure shows the Hercules WinGUI main panel:
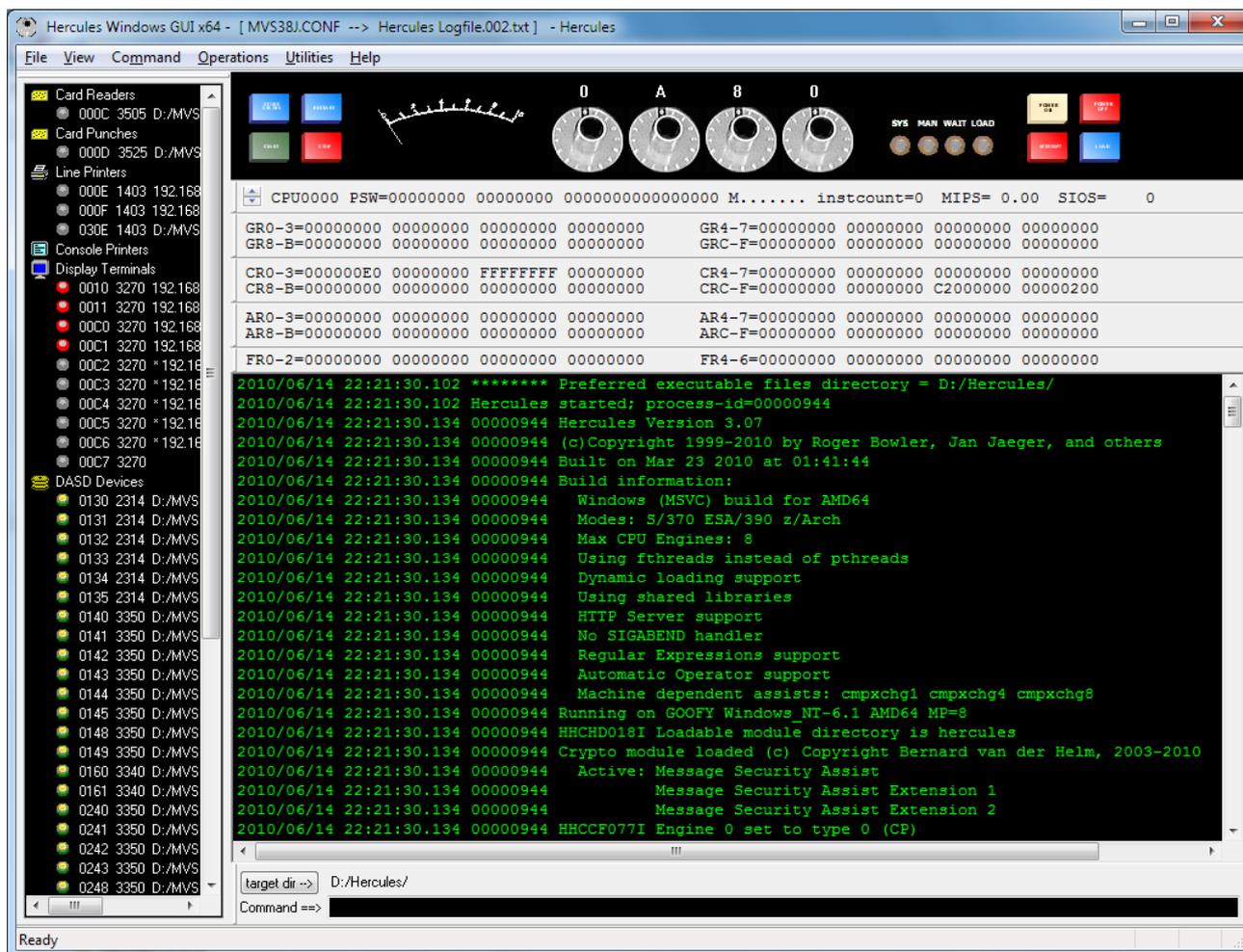
**Figure 9: Hercules Windows GUI Main Panel**

Using the GUI all of the Hercules DOS utilities are available, it is no longer necessary to know the exact syntax of each utility. Instead of having to issue cryptic command lines in native DOS such as

**HETUPD   -2   -b   D:/MVS/TAPE/TLEV009.HET   D:/MVS/TAPE/TLEV002.HET**

a standard Windows dialog box can be used to call the utility. The following figure shows the pop-up window, used to provide information to one of the Hercules utility programs (HETUPD).
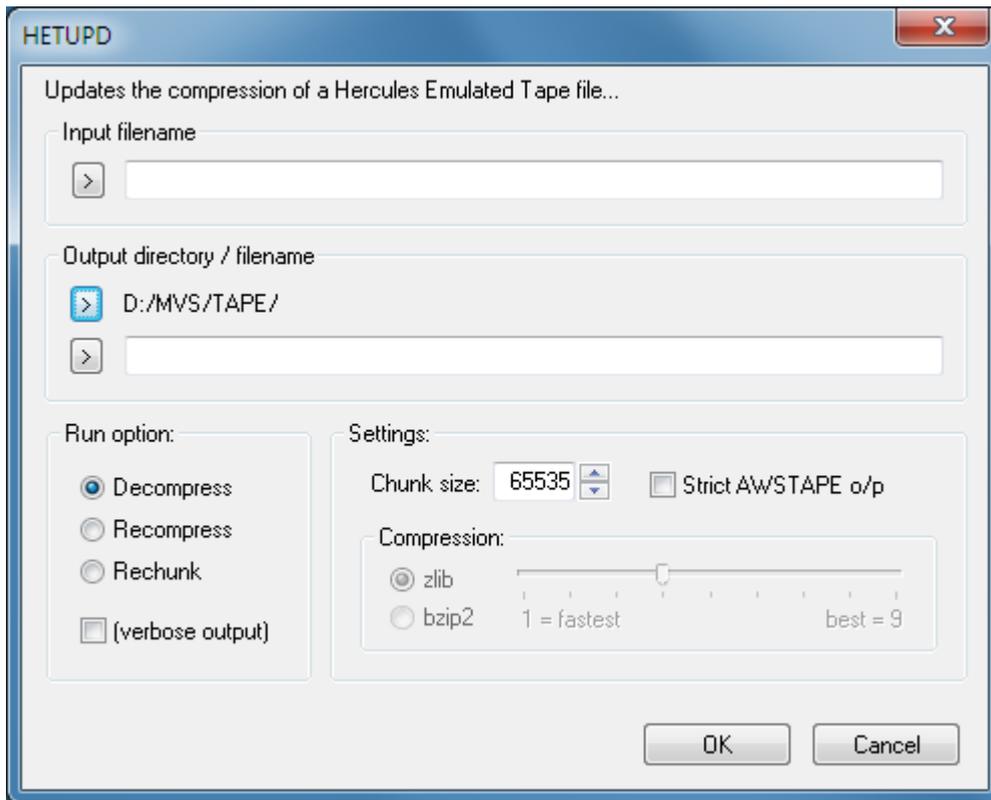
**Figure 10: Hercules Utility Window**

The WinGUI has been developed by David B. Trout (Fish). More detail about the functionality provided by the WinGUI can be found in the "User Reference Guide" and in Chapter 10 – Installing the Hercules WinGUI.

## 5.4.3 CTCI-W32

Since Hercules runs as a user process under the control of the host Windows system it does not normally have direct access to the driving system's network adapter. Until recently this presented a problem in establishing connectivity between the network and the TCP/IP stack of an operating system running un-der Hercules.

Since the development of device drivers by Fish, employing of the WinPcap's device driver, it is now possible to establish a virtual point-to-point link between the TCP/IP stack running under Hercules and Window's TCP/IP stack. This allows you to use Windows as a router to pass Ethernet frames between the Hercules TCP/IP stack and the rest of the network.

## 5.5   Additional required and optional Software

As well as the components described above other software, either required for practical use of Hercules (e.g.: tn3270 client) or that makes common tasks easier (e.g.: XMIT Manager, AWS Browse, ZZSA etc.) may be used.

### 5.5.1 tn3270 Client (required)

For virtual 3270 consoles and 3270 terminals a tn3270 client software application is required. The tn3270 client can run on the same machine as Hercules or on any Unix or Windows box with a TCP/IP connection to the Hercules machine. The supported and recommended tn3270 client for Hercules under Windows is Vista tn3270.

Vista tn3270 can be obtained from www.tombrennansoftware.com. The license fee charged by the developer of the software, Tom Brennan, is very modest. A 30 day trial version can be downloaded from his web site.

Other tn3270 clients, such as QWS3270, IBM Personal Communications, Attachmate Extra, etc., should also work in most cases. Be aware that some tn3270 clients have a bug that makes them unusable as an MVS console.

Because the tn3270 client is an independent piece of software there are no version requirements. You can use any stable release of a tn3270 client although it is recommended to always run with a current release.

### 5.5.2 XMIT Manager (optional)

The XMIT Manager is a Windows based tool that allows for the manipulation of IBM mainframe created Xmit format files. With XMIT Manager you can open Xmit files and view or extract the data within them, whether binary or text, using a graphical interface. Xmit files containing partitioned or sequential datasets are supported.

### 5.5.3 AWS Browse (optional)

The AWS Browse Utility is used to view the contents of tapes from the Windows desktop without having to start a mainframe operating system and run tape reading utilities. There are currently two implementations of AWS browse.

The original one was created by Rob Storey. The second is an enhanced version written by Fish, which is faster and has more features.

# 6.  Component Compatibility Tables

The various components under the Windows operating system that together make the full Hercules environment have some inter-dependencies on each other. It is recommended that these components be at a certain release level for every version of Hercules. The following tables list the combinations of components for several releases that have been successfully evaluated as working together.

If a combination is not listed in these tables this does not necessarily mean that it will not function. However you must asses this yourself and at your own risk. Although the listed combinations shown below have been thoroughly tested and proven to provide a stable environment, there is no guarantee that the components work in every environment and in any case.

Components that have been updated during the lifetime of a specific Hercules release are shown with their highest release level in each table. In general it is recommended to work always with the most current stable release of each component. Beta releases are not shown here. The date(s) in parenthesis show the release dates of the respective components. If there was no release of Hercules but updated components, then the word "Components" can be found before the release date.

In the case of Cygwin the release shown here is the one that Hercules was built with, rather than the highest. As Cygwin is a runtime environment, its release level must be the same or higher than the version Hercules was built with. Therefore the Cygwin release level listed for any level of Hercules is the minimum level needed for that Hercules release. Any higher level of Cygwin should also run but experience has shown that this is not always the case.

Please note that obsolete components, e.g. Cygwin, or components that were subsequently included in other packages, e.g. FishPack, TunTap32 and tt32info are mentioned in the table immediately after the release where the change occurred and are then removed from the tables.

## 6.1  Hercules V 3.09.0 (Release date: July 15, 2013)

| Component | Release |
|---|---|
| Hercules Emulator | V 3.09.0 |
| WinPcap | V 4.1.2 |
| HercGUI | V 1.11.1.5265 |
| CTCI-W32 | V 3.2.1.160 |
| FishLib | V 2.7.1.564 |
| AWS Browse | V 1.5.1.1805 |
| HercPrt | V 1.1.0.545 |

**Table 7: Hercules Release V 3.09.0 Component Compatibility Table**

## 6.2   Hercules V 3.08.1 (Release date: March 13, 2013)

| Component | Release |
|---|---|
| Hercules Emulator | V 3.08.1 |
| WinPcap | V 4.1.2 |
| HercGUI | V 1.11.1.5265 |
| CTCI-W32 | V 3.2.1.160 |
| FishLib | V 2.7.1.564 |
| AWS Browse | V 1.5.1.1805 |
| HercPrt | V 1.1.0.545 |

**Table 8: Hercules Release V 3.08.1 Component Compatibility Table**

## 6.3   Hercules V 3.08.0 (Release date: December 12, 2012)

| Component | Release |
|---|---|
| Hercules Emulator | V 3.08.0 |
| WinPcap | V 4.1.2 |
| HercGUI | V 1.11.1.5265 |
| CTCI-W32 | V 3.2.1.160 |
| FishLib | V 2.7.1.564 |
| AWS Browse | V 1.5.1.1805 |
| HercPrt | V 1.1.0.545 |

**Table 9: Hercules Release V 3.08.0 Component Compatibility Table**

## 6.4  Hercules V 3.07.0 (Release date: March 10, 2010)

| Component | Release |
|---|---|
| Hercules Emulator | V 3.07.0 |
| WinPcap | V 4.1.1 |
| HercGUI | V 1.11.1.5265 |
| CTCI-W32 | V 3.2.1.160 |
| FishLib | V 2.7.1.564 |
| AWS Browse | V 1.5.1.1805 |
| HercPrt | V 1.1.0.545 |

**Table 10: Hercules Release V 3.07.0 Component Compatibility Table**

## 6.5  Hercules V 3.06.0 (Release date: January 11, 2009)

| Component | Release |
|---|---|
| Hercules Emulator | V 3.06.0 |
| WinPcap | V 4.0.2 |
| HercGUI | V 1.11.1.5265 |
| CTCI-W32 | V 3.2.1.160 |
| FishLib | V 2.7.1.564 |
| AWS Browse | V 1.5.1.1805 |

**Table 11: Hercules Release V 3.06.0 Component Compatibility Table**

## 6.6   Hercules V 3.05.0 (Release date: June 23, 2007)

| Component | Release |
|---|---|
| Hercules Emulator | V 3.05.0 |
| WinPcap | V 4.0.1 |
| HercGUI | V 1.11.1.5265 |
| CTCI-W32 | V 3.2.1.160 |
| FishLib | V 2.7.1.564 |
| AWS Browse | V 1.5.1.1805 |

**Table 12: Hercules Release V 3.05.0 Component Compatibility Table**

## 6.7   Hercules V 3.04.0 (Release date: February 24, 2006)

| Component | Release |
|---|---|
| Hercules Emulator | V 3.04.0 |
| WinPcap | V 4.0 |
| HercGUI | V 1.11.1.5265 |
| CTCI-W32 | V 3.2.1.160 |
| FishLib | V2.7.1.564 |
| AWS Browse | V 1.5.1.1805 |

**Table 13: Hercules Release V 3.04.0 Component Compatibility Table**

## 6.8   Hercules V 3.03.1 (Release date: December 31, 2005)

| Component | Release |
|---|---|
| Hercules Emulator | V 3.03.1 |
| WinPcap | V 3.0 |
| HercGUI | V 1.9.5.4734 |
| FishPack | V 1.3.0.323 |
| TunTap32 | V 2.1.0.404 |
| tt32info | V 1.0.2.133 |
| AWS Browse | V 1.2.0.1278 |

**Table 14: Hercules Release V 3.03.1 Component Compatibility Table**

## 6.9   Hercules V 3.03.0 (Release date: December 20, 2005)

| Component | Release |
|---|---|
| Hercules Emulator | V 3.03 |
| Cygwin | **No longer needed!** |
| WinPcap | V 3.0 |
| HercGUI | V 1.9.5.4734 |
| FishPack | V 1.3.0.323 |
| TunTap32 | V 2.1.0.404 |
| tt32info | V 1.0.2.133 |
| AWS Browse | V 1.2.0.1278 |

**Table 15: Hercules Release V 3.03.0 Component Compatibility Table**

## 6.10  Hercules V 3.02.0 (Release date: December 11, 2004)

| Component | Release |
|---|---|
| Hercules Emulator | V 3.02 |
| Cygwin | V 1.5.12 |
| WinPcap | V 3.0 |
| HercGUI | V 1.8.8.4207 |
| FishPack | V 1.3.0.323 |
| TunTap32 | V 2.0.3.379 |
| tt32info | V 1.0.2.133 |
| AWS Browse | V 1.2.0.1278 |

**Table 16: Hercules Release V 3.02.0 Component Compatibility Table**

## 6.11  Hercules V 3.01.0 (Release date: November 30, 2003)

| Component | Release |
|---|---|
| Hercules Emulator | V 3.01 |
| Cygwin | V 1.5.5 |
| WinPcap | V 3.0 |
| HercGUI | V 1.6.8.3981 |
| FishPack | V 1.3.0.323 |
| TunTap32 | V 2.0.3.379 |
| tt32info | V 1.0.2.133 |

**Table 17: Hercules Release V 3.01.0 Component Compatibility Table**

## 6.12  Hercules V 3.00.0 (Release date: October 2, 2003)

| Component | Release |
|-----------|---------|
| Hercules Emulator | V 3.00 |
| Cygwin | V 1.5.5 |
| WinPcap | V 3.0 |
| HercGUI | V 1.6.8.3910 |
| FishPack | V 1.3.0.323 |
| TunTap32 | V 2.0.3.379 |
| tt32info | V 1.0.2.133 |

**Table 18: Hercules Release V 3.00.0 Component Compatibility Table**

# 7. Installation WinPcap



**Figure 11: WinPcap Logo**

## 7.1  WinPcap Packet Capture Driver

The first component you are required to install after the operating system is the WinPcap Packet Capture Driver. This can be downloaded from the website of the Politecnico di Torino (http://www.WinPcap.org) where you will also find comprehensive documentation. Save the executable in a directory of your choice.

Be sure to only use the latest version of WinPcap that is supported by CTCI-W32 supplied with the release of Hercules you are using. The most recent release of WinPcap may not be compatible with your version of Hercules. See chapter 0 (Component Compatibility Tables) for details of the supported release.

## 7.2  Installation Steps (Windows Setup)

The installation program for WinPcap creates and then initiates a kernel driver service. Please note that you need to have administrator privileges in order to create and to start the Netgroup Packet Filter (NPF) kernel driver service. You must be logged onto your Windows system as an "administrator" when you run the TESTAPP program for the first time and create the NPF service entry, or whenever you run a program that starts the NPF service. Once the NPF service has been created and / or started however, any non administrator user may use the service.

As with every installation under the Windows operating system, you should stop all running programs prior to starting the installation process. To start the installation wizard just double-click on the executable WINPCAP_v_r.EXE (where "v" is the version and "r" is the release) in the directory where you previously have saved the file.

The information screen of the WinPcap installation appears.



**Figure 12: WinPcap Setup - Information Screen**

This screen gives you an informational about WinPcap. Click on "Next >" to continue with the installation.

The welcome screen of the installation wizard appears.
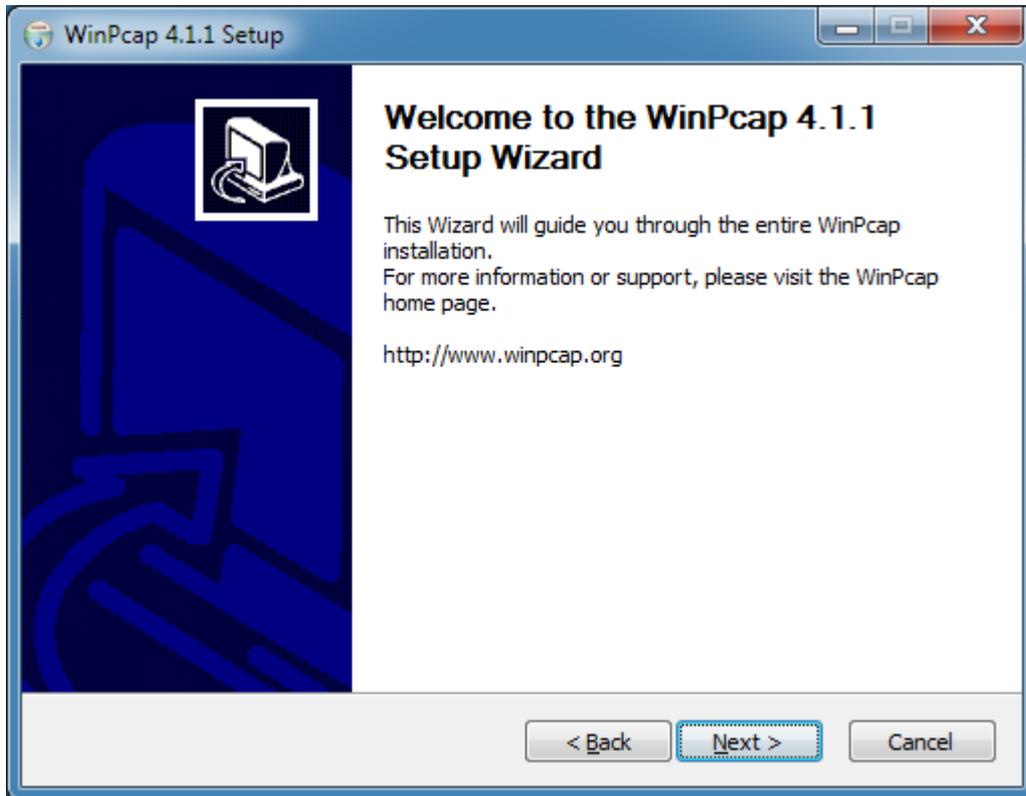


**Figure 13: WinPcap Setup - Welcome Screen**

This screen gives you the usual information presented when starting installation programs under Windows. Click on "Next >" to continue with the installation.

On the following screen you are presented with the license agreement. Read the license agreement carefully. You have to accept the license agreement in order to continue with the installation process.
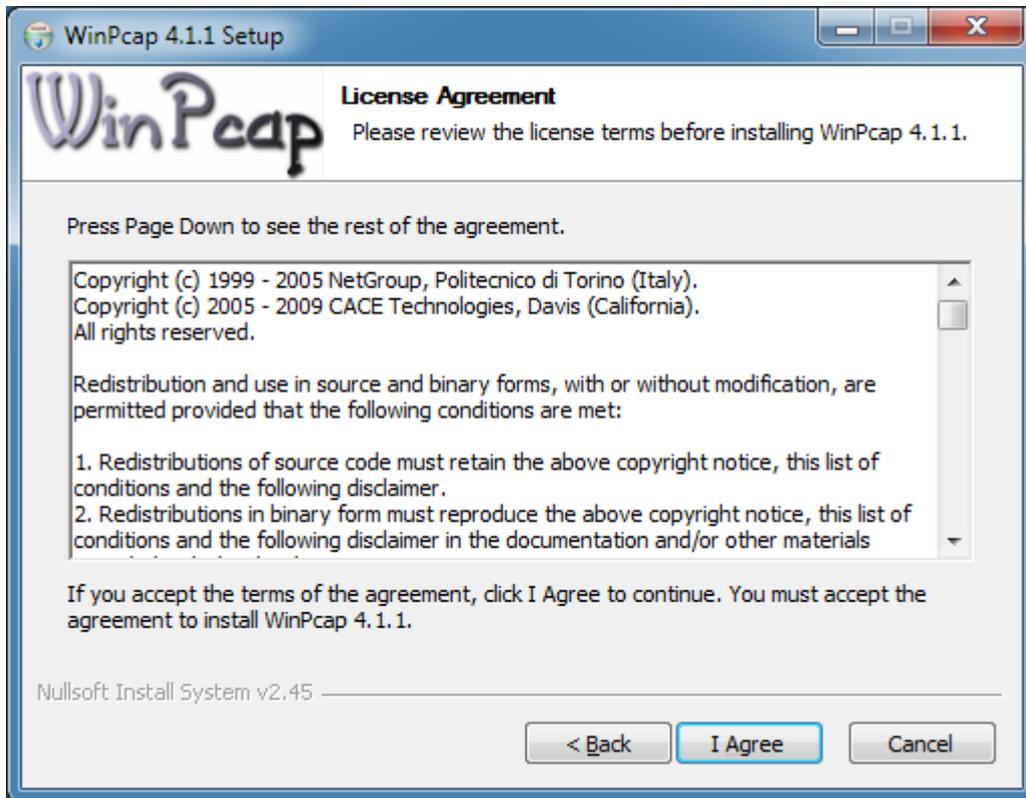


**Figure 14: WinPcap Setup - License Agreement**

After reading the license agreement, accept it by clicking on the "I Agree" button to continue the installation process.

Next the setup program presents some installation options. Be sure to have selected "Automatically start the WinPcap driver at boot time" option.
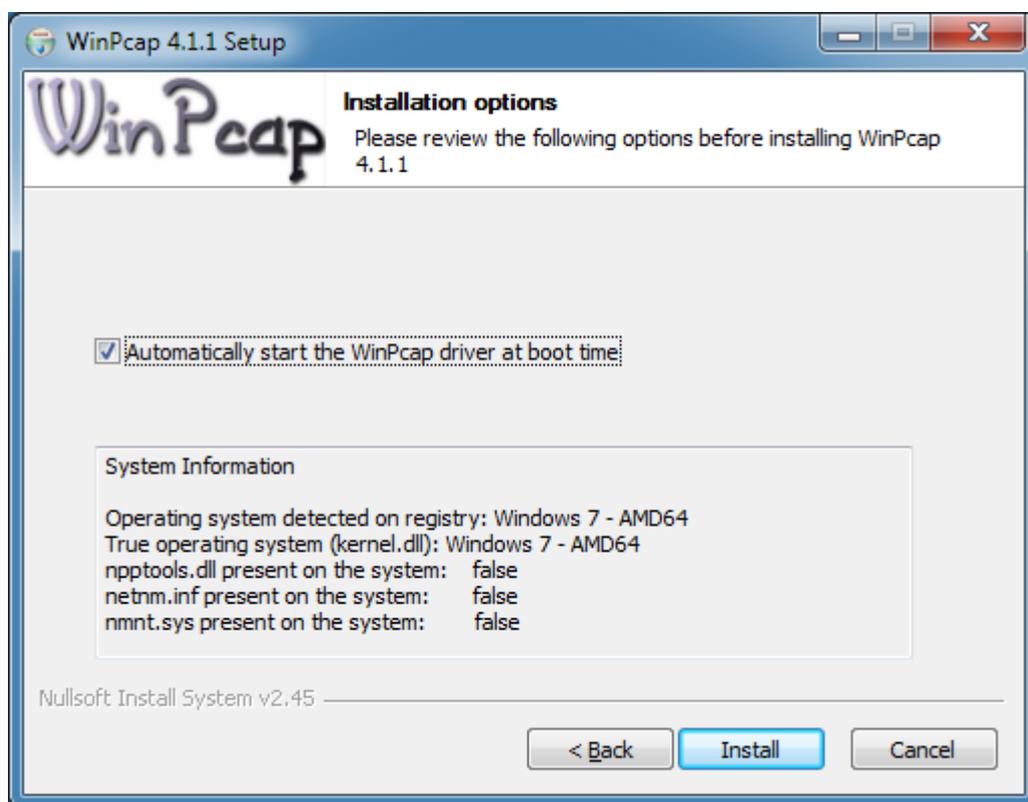


**Figure 15: WinPcap Setup - Installation Options**

The installation process now automatically begins. You do not have to select a target directory and there are no other decisions to make. A progress bar in a new window will appear. On a fast machine it is possible that the progress window just flickers for a short time, or that you will not see the window at all as the installation process is very fast.
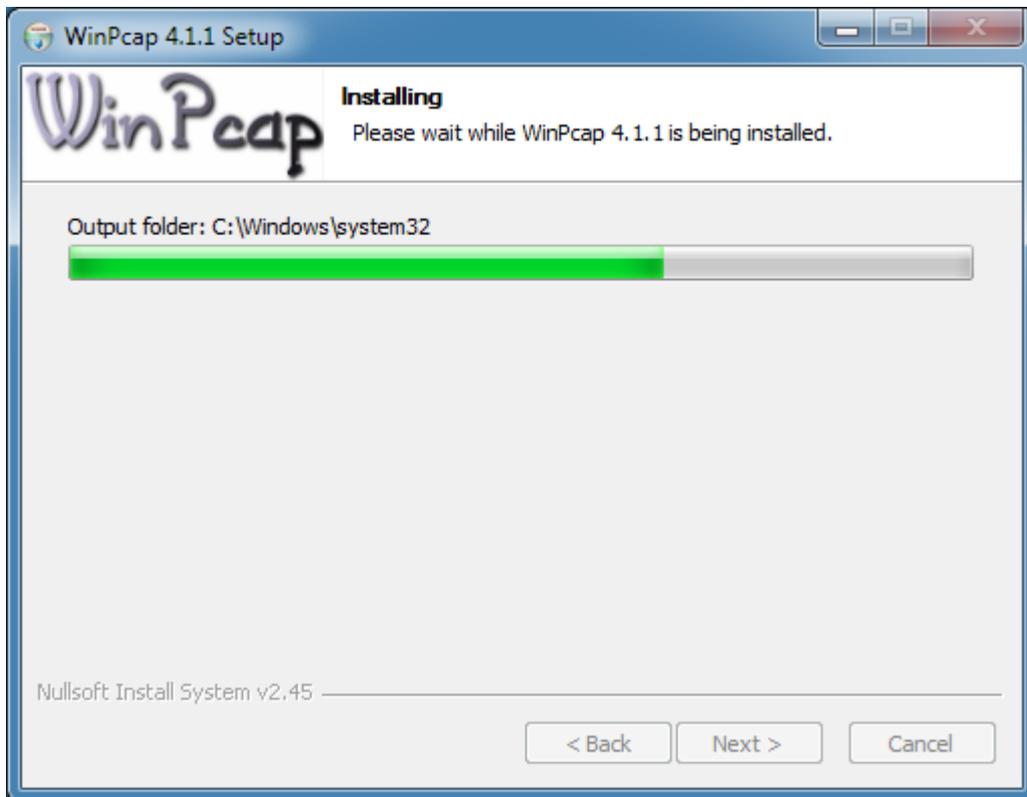


**Figure 16: WinPcap - Setup Status**

As soon as all files are copied to their default destination and the installation process is complete the final screen shown below is displayed.
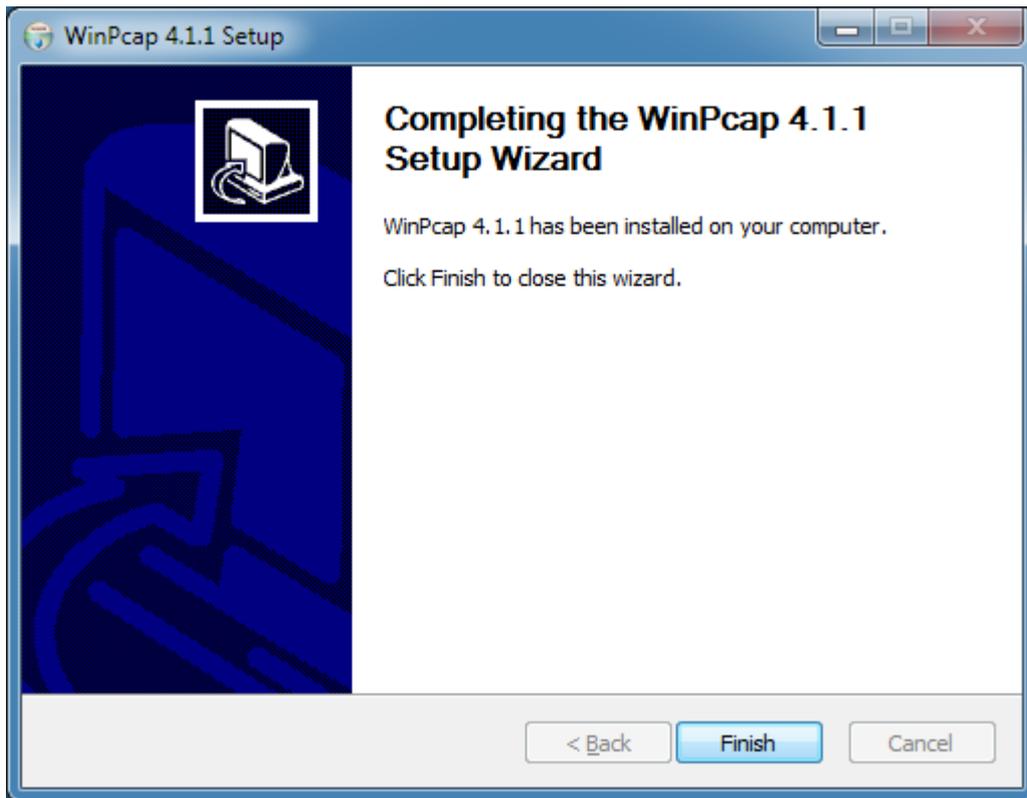
**Figure 17: WinPcap Setup - Installation Complete**

If an older version of WinPcap was installed on the machine it is strongly recommended that you reboot the system. Click on "Finish" to complete the installation process, then continue with the customization steps.

# 8. Installing the Hercules Emulator

## 8.1 Downloading the Binaries

The "ready-to-run" binaries can be downloaded from http://www.hercules-390.eu. For Windows users there are currently two varieties available:

- MSVC Windows installer package (filename "hercules-*v.rr*-w64.msi" or "hercules-*v.rr*-w64.msi")

- MSVC binaries only archive (filename "hercules-*v.rr*-w64.zip" or "hercules-*v.rr*-w32.zip")

Where *'v.rr'* in the filename specifies the version and release of the Hercules Emulator contained.

The difference between these packages and how each is installed are described in the following sections.

## 8.2 Choosing a Package

Which of these packages is the right one for you depends on your needs. The former Cygwin package is no longer available because since Hercules release 3.03.0 Cygwin is no longer a pre-requisite.

The new MSVC packages, the self-extracting archive (.exe) and the Windows installer package (.msi), are very similar. The main difference being that the Windows installer package provides the standard Windows installation dialog for both installation and removal of the emulator. The Windows installer package also creates Start Menu entries for both the documentation and to start the DOS command prompt in the Hercules binary directory.

Some people want to have the full control over the whole installation process and therefore prefer the self-extracting archive (.exe). On the other hand however, some people prefer to have a simple installation routine, similar to other Windows software packages.

***Note:*** Installing the .msi Windows Installer package ensures the required Microsoft Runtime components are installed and also provides convenience shortcuts in the programs menu. If the required components are already present and the shortcuts are not needed on the target system, the self-extracting or .zip archive may be used instead.

## 8.3 Installation Steps (MSVC Windows Installer Package)

The "MSVC Windows Installer Package" is a standard Windows installation dialog as used for most Windows application installations. Using this method the 'C' runtime library files are installed if needed in a separate directory structure under the Windows system directory (i.e. "%windir%\WinSxS").

This new Windows technique (SxS means "Side by Side") allows multiple versions of the same DLL to coexist in the same system. For example should application "A" need foobar.dll version 1.12 and application "B" needs foobar.dll version 1.14, both DLLs can coexist in the same system.

The following installation is shown with the 64-bit version of Hercules but applies also to the 32-bit version.

To start the installation dialog run the .MSI executable file. A welcome window is presented first.
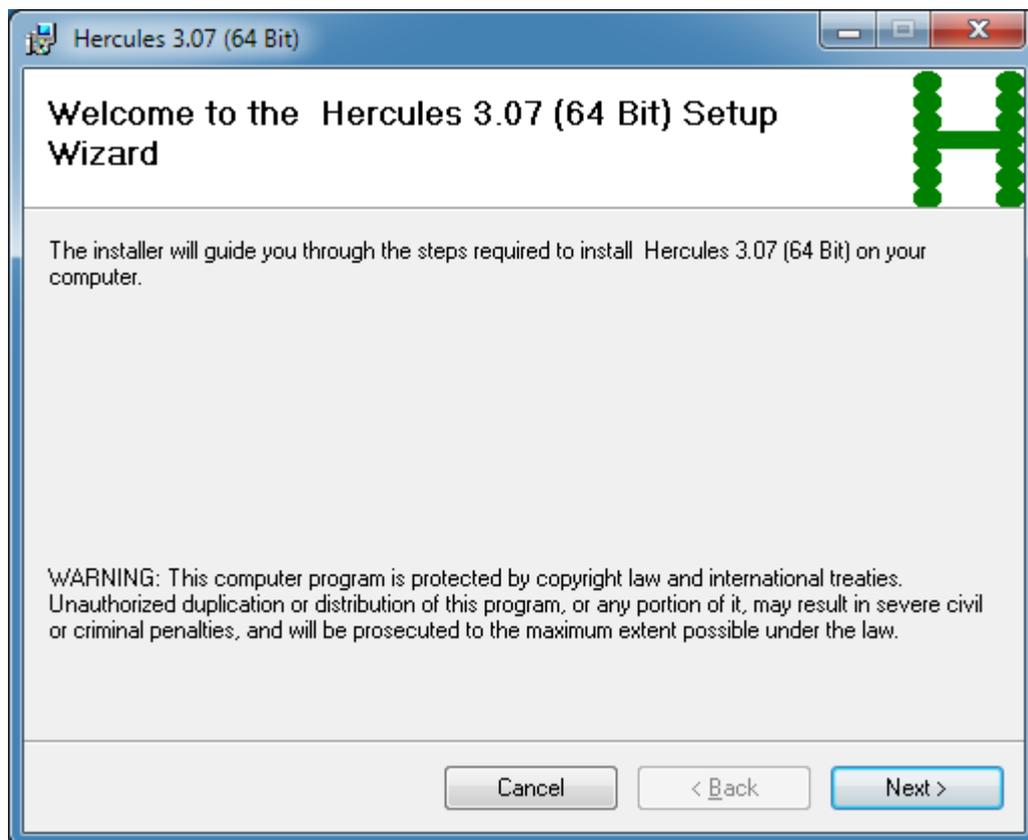


**Figure 18: Welcome Window (MSVC Installer Package)**

Clicking on "Next" continues the installation process.

The next window lets you specify the installation folder. You can use the default or specify any directory of your choice.
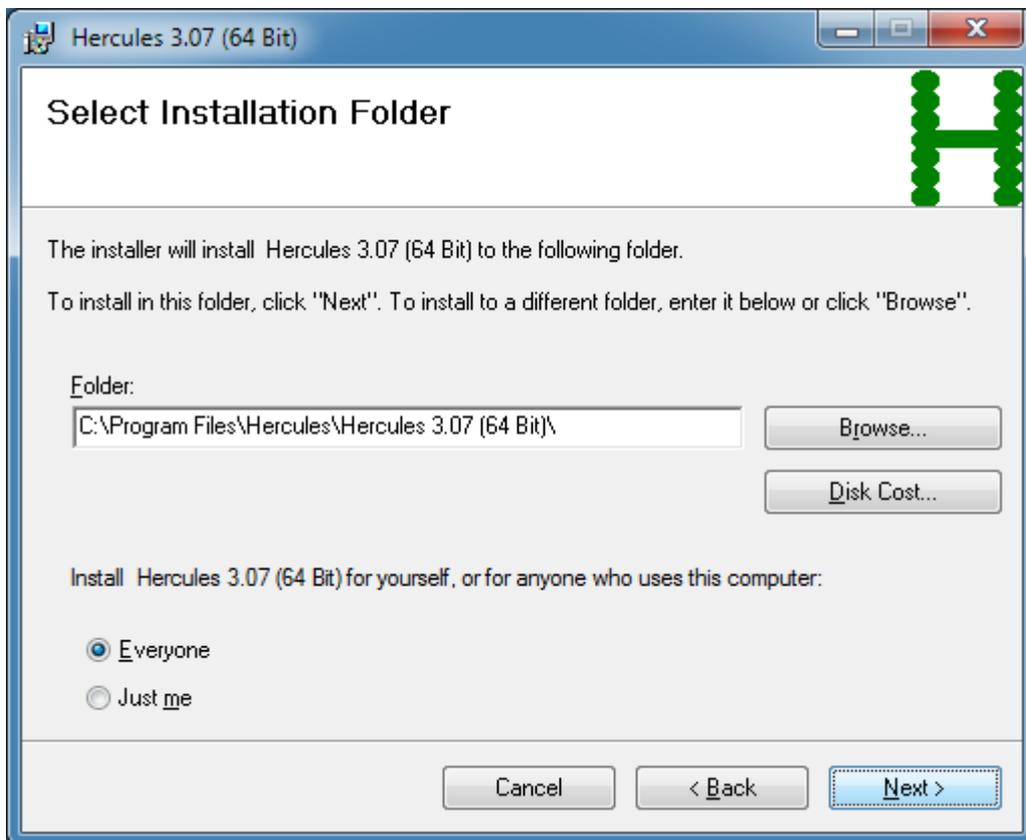


**Figure 19: Installation Directory Selection (MSVC Installer Package)**

If you plan to run different Hercules versions in parallel then the default naming convention which contains the version and release levels in the directory name is useful. If your intention is to run one Hercules version at a time then you may prefer to shorten the directory name to simply "Hercules" as in the example above.

Before installing the package you can examine your available disk space by clicking on the button "Disk Cost". This will display all available disk partitions on which Hercules can be installed and shows the disk space before and after installing the package. Hercules itself, without a mainframe operating system installed, uses approximately 11 MB of disk space.
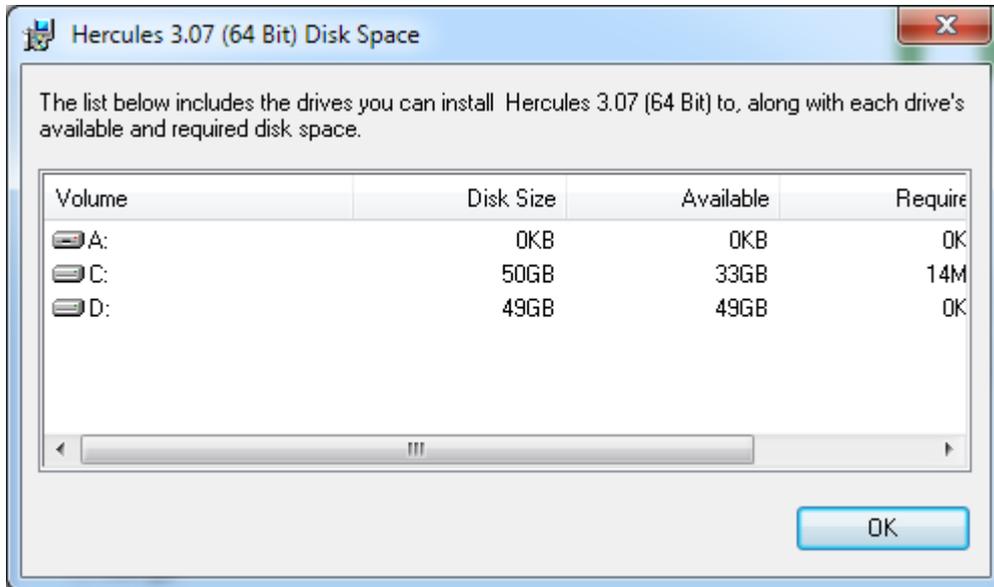


**Figure 20: Disk Space Information (MSVC Installer Package)**

When you are satisfied with the installation options, click on "OK".

A confirmation dialog is presented, this is a final opportunity to change any of your selections made in the previous screens.
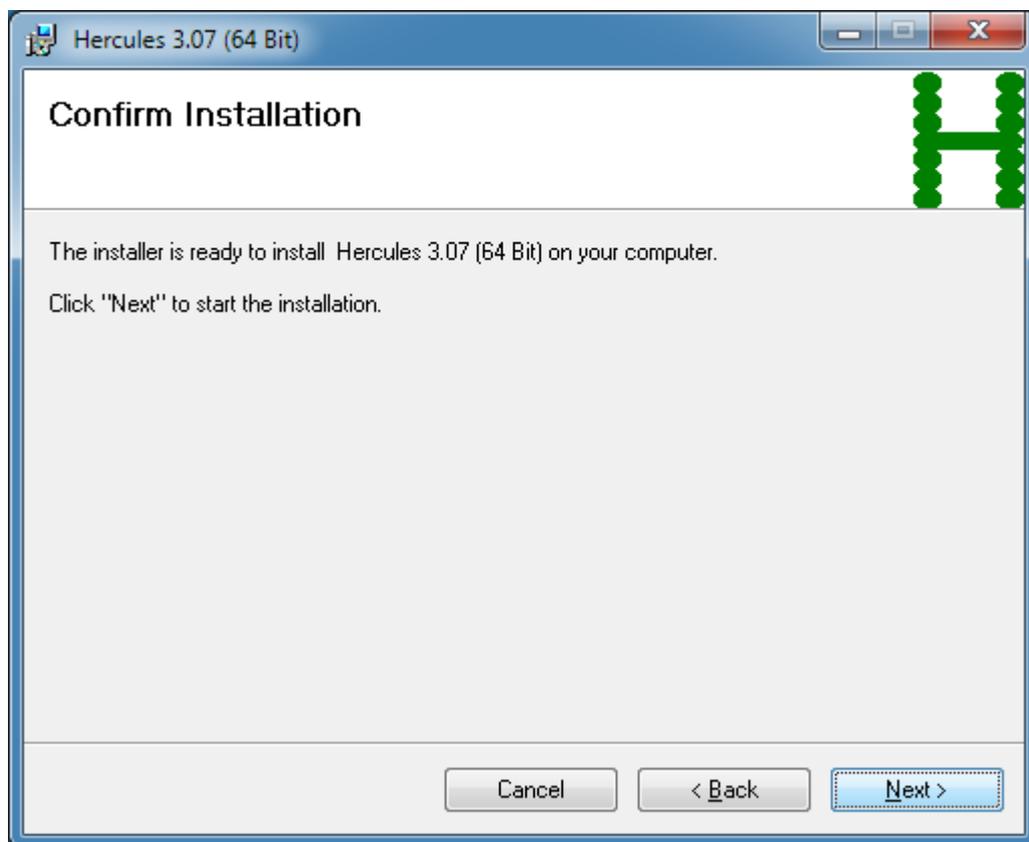


**Figure 21: Installation Confirmation (MSVC Installer Package)**

If you do not want to change any of your choices, click on "Next >" to start the actual installation process.

The installer now begins to copy files to the destination directories.
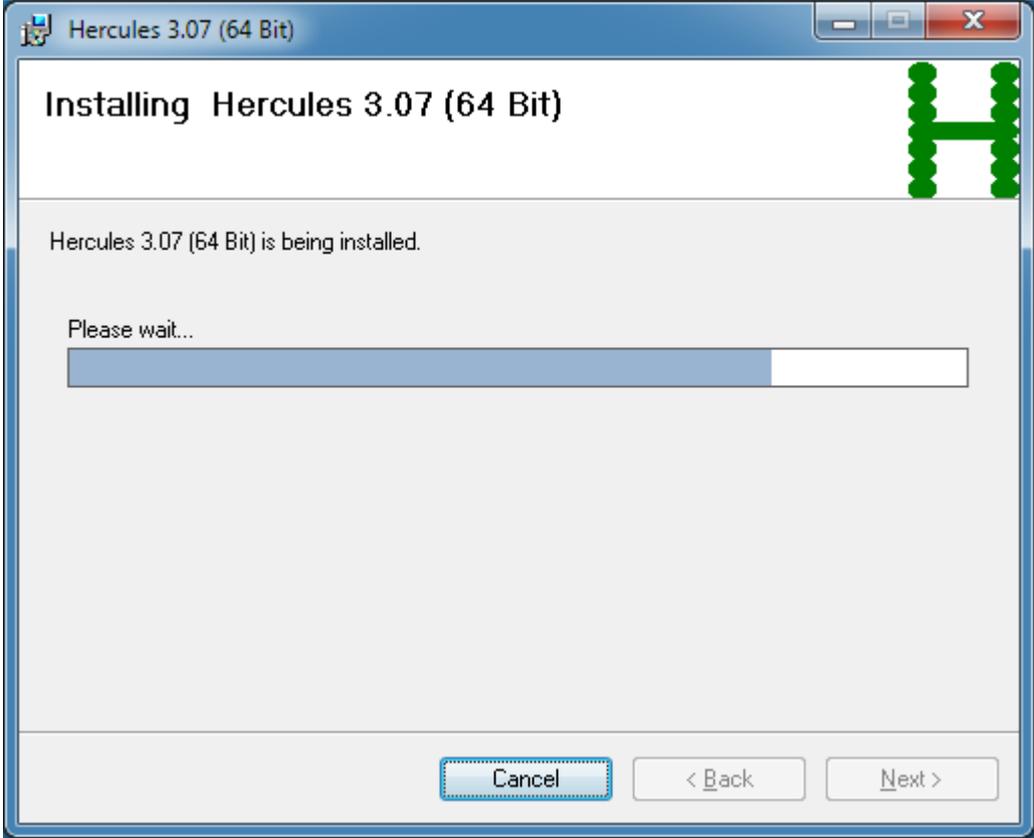


**Figure 22: Installation Progress Bar (MSVC Installer Package)**

If necessary the process of copying files can be stopped by clicking on "Cancel".

After a few seconds the installation process will finish and present the final window.
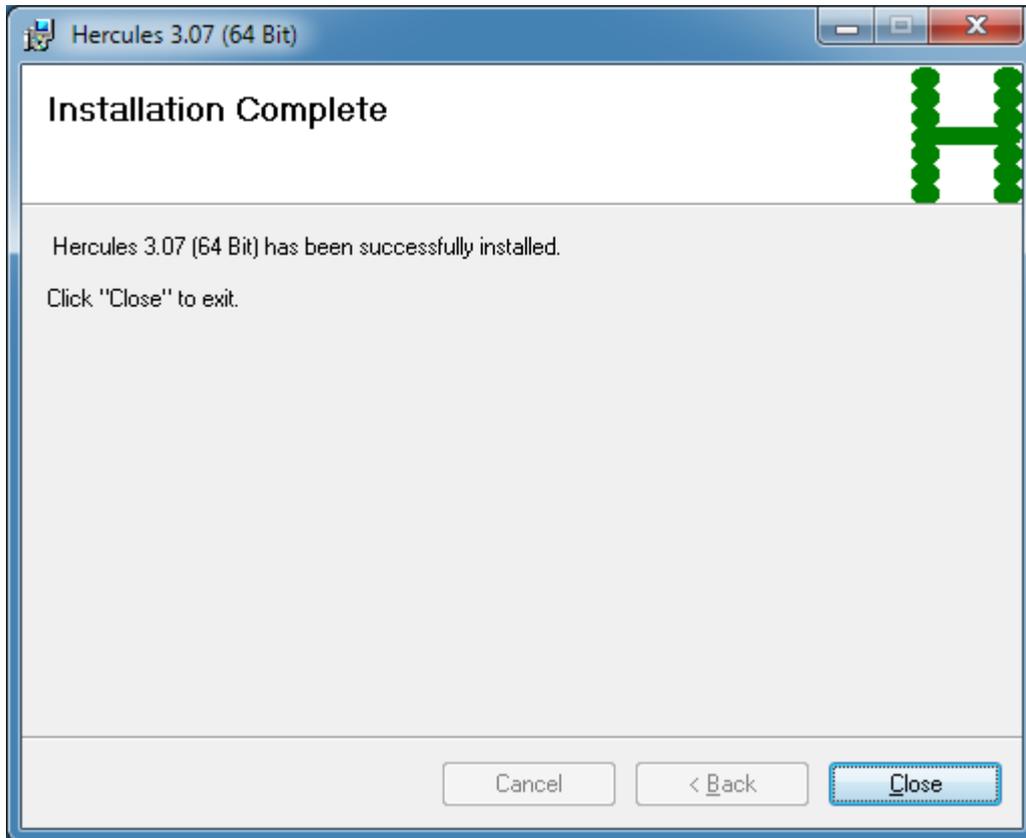


**Figure 23: Installation Complete (MSVC Installer Package)**

Click on "Close" to terminate the installation dialog and proceed to customise your installation as described in the following sections.

## 8.4   The Microsoft Windows Installer

The Microsoft Windows Installer is an installation and configuration service provided with Windows. The executable program that interprets packages and installs products in a Windows environment is called msiexec.exe. The general syntax of msiexec is as follows:

```
msiexec /option <required> [optional]
```

| | |
|---|---|
| **msiexec** | This is the name of the microsoft Windows Installer. |
| **/option** | This specifies the options as described in the following sections. |
| **<required>** | Required parameter of the specified option. |
| **[optional]** | Optional parameter of the specified option. |

For more information about syntax and options of the Microsoft Installer Packages than described in these sections please consult the Microsoft Installer SDK documentation. Please note that not all switches are useful for Hercules packages.

## 8.4.1  Install Options

These are the install options:

```
</package | /i> <Product.msi>
/a <Product.msi>
/j<u | m> <Product.msi> [/t <TransformList>] [/g <LanguageID>]
</uninstall | /x> <Product.msi | ProductCode>
```

| | |
|---|---|
| **/package** | Install or configure a product. |
| **/i** | Same as "/package". |
| **/a** | Administrative install - install a product on the network. |
| **<Product.msi>** | This is the name of the installer package (i.e."hercules-3.07-w32.msi"). |
| **/j** | Advertise a product: |
| | **u** – To the current user. |
| | **m** – To all users. |
| **/t** | Applies a the transform list to an advertised product (see below). |
| **<TransformList>** | Specifies the transform list. |
| **/g** | Set the language ID (see below). |
| **<LanguageID>** | This specifies the language ID. To determine the value for languages, consult Microsofts Platform SDK documentation. |
| **/uninstall** | Uninstall a product. |
| **/x** | Same as /uninstall. |
| **<ProductCode>** | This is the GUID (Globally Unique Identifier). The GUID is the principal identification of an application or product. |

## 8.4.2  Display Options

These are the display options:

```
/help /quiet /passive /q[n | b | r | f]
```

| | |
|---|---|
| **/help** | Display help information. |
| **/quiet** | Quiet mode, no user interaction required. |

| /passive | Unattended mode, display only the progress bar. |
|---|---|
| /q | Sets the user interface level: |

        **n** – No user interface.

        **b** – Basic user interface.

        **r** – Reduced user interface.

        **f** – Full user interface (default).

## 8.4.3  Restart Options

These are the restart options:

```
/norestart /promptrestart /forcerestart
```

| **/norestart** | Do not restart the computer after the installation is complete. |
|---|---|
| **/promptrestart** | Prompts the user for a restart of the computer if necessary. |
| **/forcerestart** | Always restart the computer after the installation is complete. |

## 8.4.4  Logging Options

These are the logging options:

```
/l[i | w | e | a | r | u | c | m | o | p | v | x | + | ! | *] <LogFile>
/log <LogFile>
```

| **/l** | Specifies the logging options. |
|---|---|

        **i** – Status messages.

        **w** – Nonfatal warnings.

        **e** – All error messages.

        **a** – Startup of actions.

        **r** – Action-specific records.

        **u** – User requests.

        **c** – Initial UI parameters.

        **m** – Out-of-memory or fatal exit information.

        **o** – Out-of-disk-space messages.

        **p** – Terminal properties.

        **v** – Verbose output.

**x** – Extra debugging information.

**+** – Append to existing log file.

**!** – Flush each line to the log file.

**\*** – Log all information, except for 'v' and 'x' options.

**\<LogFile\>**            Name and optional path of the log file.

**/log \<LogFile\>**       Equivalent of "/l* \<LogFile\>".


## 8.4.5  Update Options

These are the update options:

```
/update <Update1.msp>[;Update2.msp]
/uninstall <PatchCodGuid>[;Update2.msp] /package <Product.msi | ProductCode>
```

**/update**              Apply updates.

**\<Update.msp\>**       This is the name of the Windows installer patch (not relevant for Hercules because Hercules does not provide installer patches).

**/uninstall**           Remove updates for a product.

**\<PatchCodGuid\>**     This is the GUID (Globally Unique Identifier) of the patch.

**/package**             Specifies the package to be updated (see below).

**\<Product.msi\>**      This is the name of the installer package (i.e."hercules-3.07-w32.msi").

**\<ProductCode\>**      This is the GUID (Globally Unique Identifier). The GUID is the principal identification of an application or product.


## 8.4.6  Repair Options

These are the installer package repair options:

```
/f[p | o | e | d | c | a | u | m | s | v] <Product.msi | ProductCode>
```

**/f**                   Specifies the repair options. Repairs a product:

**p** – Only if a file is missing.

**o** – If a file is missing or an older version is installed (default).

**e** – If a file is missing or an equal or an older version is installed.

**d** – If a file is missing or a different version is installed.

**c** – If a file is missing or the checksum does not match the calculated value.

**a** – Forces all files to be reinstalled.

---

$\quad$ **u** – All required user-specific registry entries (default).

$\quad$ **m** – All required computer-specific registry entries (default).

$\quad$ **s** – All existing shortcuts (default).

$\quad$ **v** – Runs from source and recaches local package.

**\<Product.msi\>**$\quad\quad$This is the name of the installer package (i.e."hercules-3.07-w32.msi").

**\<ProductCode\>**$\quad\quad$This is the GUID (Globally Unique Identifier). The GUID is the principal identification of an application or product.

## 8.5  Installation Steps (MSVC Binaries Only Archive)

The MSVC native version of the Hercules Emulator is delivered as a ZIP archive. In this archive the binaries and all other necessary files (e.g. html files etc.) are placed in a predefined directory structure required to run Hercules.

Extracting these files does not invoke the Windows installer therefore will not install the "Microsoft Visual Studio 8 'C' Runtime Library" as a system file. Please be sure to have these libraries installed on your system.

To extract the Hercules files from the archive just run WinZip or another software which is able to extract ZIP-files. Extract the files to a folder of your choice. The necessary directory structure is created auto-matically.

The root directory that is created is called "hercules-*v.rr* (64 Bit)" or "hercules-*v.rr* (32 Bit)", where *v* does stand for the current version and *rr* for the current release. This naming convention is useful if you plan to run several Hercules versions in parallel.  If your intention is to run only one Hercules version at a time the directory can be renamed to "Hercules" without a version or release indication.

Next proceed to section 8.6 'Customisation Steps' below.

## 8.6  Customization Steps

After the actual installation is complete there are additional customization steps required, these include:

- Creating the Hercules Configuration File
- Creating the Hercules Startup Batch File
- Creating the Hercules Run-Commands File
- Creating the Terminal Batch File

These manual customization steps are explained in detail over the following sections.

### 8.6.1  Creating the Hercules Configuration File

When starting the Hercules Emulator from either a DOS command line or via the Hercules Windows GUI, you may specify the name of a configuration file as a parameter:

HERCULES [ -f filename ] …

HERCGUI [ -f filename ] …

where *filename* is the name of the configuration file. The default filename if none is specified during the startup is 'hercules.cnf'. The name of the default configuration file may be overridden via the environment variable HERCULES_CNF.

The configuration file is an ASCII text file that is used to describe the processor definition, the device layout and any runtime parameters. Details of the format and acceptable directives that can be made within the file are found in Chapter 3 of the Hercules User Reference Guide.

## 8.6.2 Creating the Hercules Startup Batch File

Although the Hercules Emulator can be started manually from a DOS command prompt it is often easier to establish a batch file to do this. The batch file can then be executed by double-clicking it from Windows Explorer, called directly from a DOS command prompt window or it can be integrated as a shortcut into the Windows start menu or desktop.

### 8.6.2.1 Hercules Startup Batch File

The following figure shows an example of a Hercules startup batch file using the native Hercules console.

```
@ECHO OFF
REM
REM *************************************************************************************
REM CHECK / SET HERCULES PATH                                                       (1)
REM *************************************************************************************
REM
IF %SETHERC%.==1. GOTO RUNIT
SET SETHERC=1
PATH D:\Hercules;%PATH%
SET HERCULES_RC=D:\MVS\CONF\HERCULES.RC
:RUNIT
REM
REM *************************************************************************************
REM START HERCULES EMULATOR                                                         (2)
REM *************************************************************************************
REM
START D:\Hercules\Hercules -f D:/MVS/CONF/MVS38J.CONF >D:\MVS\LOG\Hercules_Log.txt
EXIT
```

**Figure 24: Hercules Startup Batch File**

This example batch file contains two main sections:

(1) In the first section a check is made to determine if the path to the Hercules binaries has already been set. If so, then the rest of the first section is skipped. If not, then the path to the Hercules binaries is set and the environment variable %SETHERC% is set to '1' to indicate that the path has been successfully set.

(2) In the second section the Hercules Emulator is started with a configuration file located in a directory other than the emulator itself.  Additionally the path and filename for the log file is set.

### 8.6.2.2  Hercules Windows GUI Startup Batch File

The following figure shows an example of a Hercules startup batch file using the Hercules Windows GUI.

```
@ECHO OFF
REM
REM ***********************************************************************
REM CHECK / SET HERCULES PATH                                          (1)
REM ***********************************************************************
REM
IF %SETHERC%.==1. GOTO RUNIT
SET SETHERC=1
PATH D:\Hercules;%PATH%
SET HERCULES_RC=D:\MVS\CONF\HERCULES.RC
:RUNIT
REM
REM ***********************************************************************
REM START HERCULES WIN GUI AND HERCULES EMULATOR                       (2)
REM ***********************************************************************
REM
START D:\Hercules\HercGui -f D:/MVS/CONF/MVS38J.CONF
EXIT
```

**Figure 25: Hercules Windows GUI Startup Batch File**


This example batch file contains two main sections:

(1)  In the first section a check is make to determine if the path to the Hercules binaries has been al-
ready set. If so, then the rest of the first section is skipped. If not, then the path to the Hercules
binaries is set and the environment variable %SETHERC% is set to '1' to indicate that the path
has been successfully set.

(2)  In the second section the Hercules Windows GUI is started which will in turn start the Hercules
Emulator, using a configuration file located in a separate directory to the emulator itself.
Differently to the first example above, the filename and path of the log file are not given as a
parameter to the Windows GUI, rather they are configured within the GUI itself.


## 8.6.3  Creating the Hercules Run-Command File

Hercules also provides the ability to automatically execute Hercules panel commands after startup via the
'run-commands' file. If the run-commands file exists when Hercules starts, each line contained in the file
is read and interpreted as panel command.

In the following example file the Windows telnet program is started in a shell, all currently valid Hercules panel commands are displayed and MAXRATES is reset. Finally a couple of tn3270 sessions for consoles and TSO terminals are started.

```
sh startgui telnet localhost 3270

?

maxrates

sh startgui C:\Programs\Vista32\vista32.exe MVS_MST.ses

pause 3

sh startgui C:\Programs\Vista32\vista32.exe MVS_OPR.ses

pause 3

sh startgui C:\Programs\Vista32\vista32.exe MVS_TSO.ses

pause 3

sh startgui C:\Programs\Vista32\vista32.exe MVS_TSO.ses

pause 3

sh startgui C:\Programs\Vista32\vista32.exe MVS_TSO.ses
```

**Figure 26: Hercules Run-Commands File**

If the "sh" (shell) command is used in the run-commands file, then the "startgui" keyword should also be used when starting external applications if one of the following is true:

a) If starting Windows GUI applications (as opposed to command-line programs), regardless whether or not the Hercules Windows GUI is being used.

b) If starting any program, Windows GUI programs or command-line programs (either via the Hercules panel command-line or via the run-commands file), when running Hercules via the Hercules Windows GUI.*

* If you wish to capture the output of a shell command however (e.g. "sh dir"), then you should not use startgui since it prevents the output from being captured / piped back to Hercules or the Hercules Windows GUI.

Creative use of the run-commands file can completely automate Hercules startup and initiate the IPL of the mainframe operating system. For details about the usage of the run-commands file see the "Hercules User Reference Guide".

## 8.6.4 Creating the Terminal Batch File

When the Hercules Emulator is been started using one of the previously described batch files, terminals (consoles and terminals) must then be connected. If they are not automatically started via the rc-file as described in the section above, the easiest way to do this is also with a separate batch file, which can start several instances of a terminal emulation program.

In the example file below several terminals are connected to the Hercules emulator using tn3270 emulation software "Vista tn3270" from Tom Brennan. The initiation of terminal connections to Hercules is particular to the 3270 emulator you chose to use – consult the documentation of your emulator software for these details.

```
@ECHO OFF
REM
REM *********************************************************************
REM Start master console (0700) and wait for connection to Hercules     (1)
REM *********************************************************************
REM
START "C:\PROGRAMS\VISTA32\VISTA32.EXE C:\PROGRAMS\VISTA32\MVS_MST.SES"
PAUSE 2
REM
REM *********************************************************************
REM Start operator console (0701) and wait for connection to Hercules    (2)
REM *********************************************************************
REM
START "C:\PROGRAMS\VISTA32\VISTA32.EXE C:\PROGRAMS\VISTA32\MVS_OPR.SES"
PAUSE 2
REM
REM *********************************************************************
REM Start TSO terminals (0702-0704) and wait for connection to Hercules  (3)
REM *********************************************************************
REM
START "C:\PROGRAMS\VISTA32\VISTA32.EXE C:\PROGRAMS\VISTA32\MVS_TSO.SES"
PAUSE 2
START "C:\PROGRAMS\VISTA32\VISTA32.EXE C:\PROGRAMS\VISTA32\MVS_TSO.SES"
PAUSE 2
START "C:\PROGRAMS\VISTA32\VISTA32.EXE C:\PROGRAMS\VISTA32\MVS_TSO.SES"
EXIT
```

**Figure 27: Terminal Batch File**


This example terminal batch file contains three sections:

    (1)  Connecting the master console

    (2)  Connecting the alternate (operator) console

    (3)  Connecting the TSO terminals


Generally these three sections are identical except for the session profile used. The number of terminals to be started depends on your needs although it is recommended that at least three terminal sessions (master console, alternate console and TSO terminal) be started.

The PAUSE command is used to define a two second delay and helps ensure that the terminal sessions connect in the correct sequence to the intended addresses. The length of this wait interval depends on the speed of the host system CPU and should be changed to a value appropriate for your environment. You may need to experiment with this setting in order to discover an appropriate value.

# 9.   Installing the Hercules Windows GUI

## 9.1   Downloading the Binaries

The Hercules Windows GUI can be downloaded from www.softdevlabs.com. Several other programs from the developer (David B. Trout, aka 'Fish') require a custom DLL, FishLib.dll, it may be appropriate to download the FishLib package as well.

The GUI also requires some Microsoft Foundation Classes (MFC) and C runtime DLLs (MFC42.DLL, MSVCRT.DLL, MSVCP60.DLL and DbgHelp.DLL). It is possible that these DLLs are already present in the Windows system directory. If not though, they must be downloaded and copied to the Windows system directory. These DDLs are all available from www.softdevlabs.com.

The source code for the Hercules Windows GUI is no longer available due to current efforts to commercialise this product.

**Note:** Beginning with release 1.11.1.5265 of the Windows GUI additional DLLs are required. These are Microsoft MFC and VC Runtime DLLs that you can download from the address mentioned above. The installation takes a few seconds and does not require a reboot. There are a 32-bit and a 64-bit version of these DLLs. Please ensure you are using the correct one according to the product you are installing (32-bit or 64-bit version of the Windows GUI).

These DLLs are normally automatically installed as part of a standard Microsoft Installer program (.msi), however as the Windows GUI does not have an installer program you will need to do this manually. Note that you only need to install these C Runtime DLLs once even if new versions of HercGUI are subsequently installed.

## 9.2   Installation Steps

The installation involves simply unzipping the executables into the directory where the Hercules executables reside. No other installation steps are required. Uninstalling is the reverse, just delete the files.

On its first startup the Hercules Windows GUI adds a new key to the Windows registry to hold some configuration settings. To completely remove all trace of the GUI you must manually delete this key using a registry editor such as the Microsoft "regedit" utility.

The Hercules Windows GUI uses the following registry key:

"HKEY_CURRENT_USER/Software/Software Development Laboratories/Hercules"

## 9.3   Customization Steps

The first time the Hercules Windows GUI is started it will open the Preferences dialog. Completing this dialog is the only customization required, it adds necessary information (e.g. directory names) required for the GUI to operate.

The following topics describe the various dialogs in detail. Please note that this installation section describes many of the tasks involved in the use of the GUI in preference to separating this information into the Hercules User Reference Manual.

## 9.4 Main Screen

The various panels of the main screen can be re-located to suit the user's preferences. The device list bar can be docked on either side of the screen and the various status bars at either the top or bottom of the screen. You can float them in a window by themselves or you can hide them altogether.

To do this just grab the panel using your mouse and drag it to your preferred location. The screen layout is remembered across sessions. If you place the controls panel at the bottom of the screen, then it will be in the same place the next time you start Hercules.
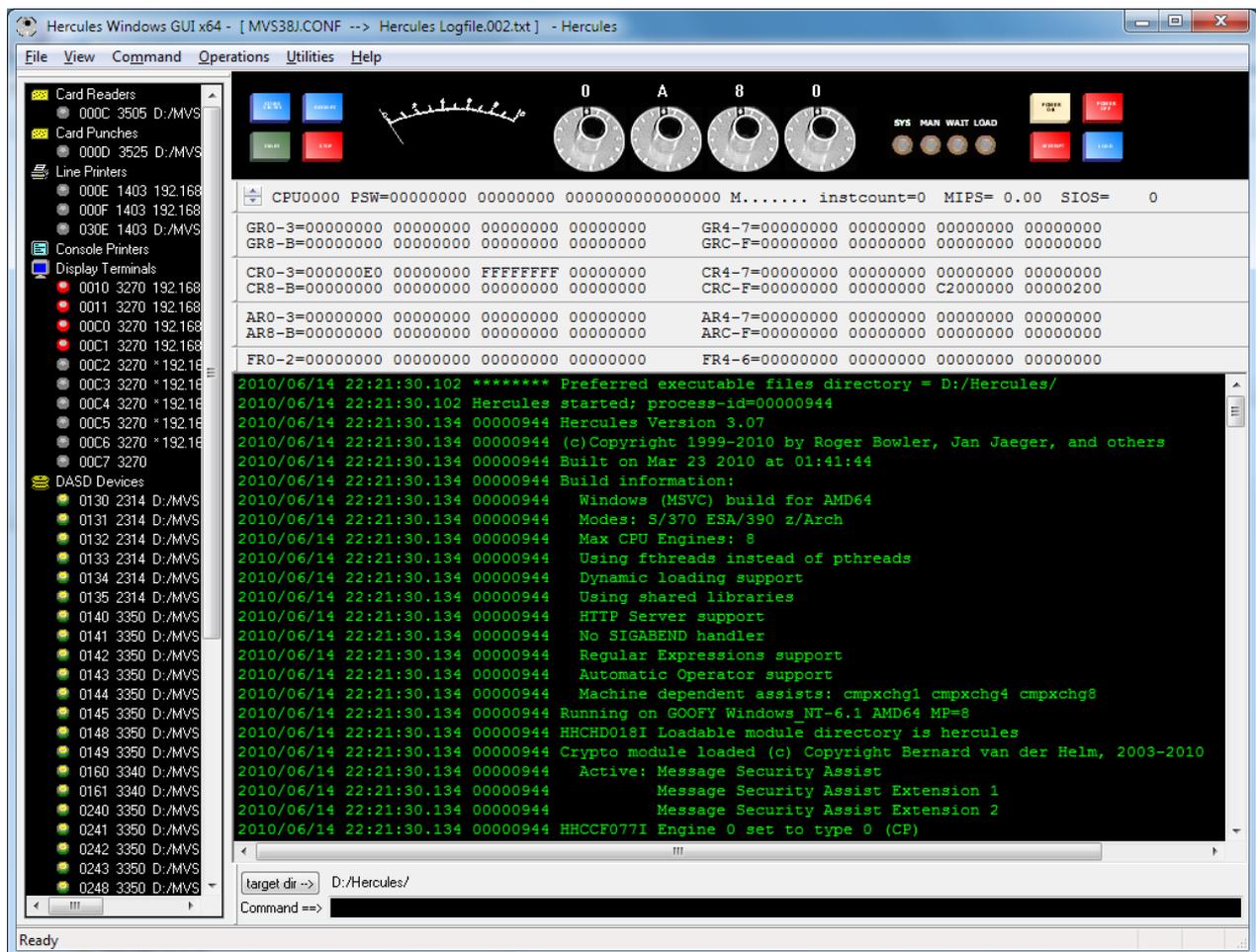


**Figure 28: Hercules Windows GUI Main Panel**

Since version 1.9.5 the Hercules Windows GUI allows you to specify the target directory that the 'sh' (shell) command will use. Simply browse to and select the desired directory using the GUI.

Once this is set any shell command entered will be processed using the defined directory as its current working directory. This provides a workaround for the fact that the Windows GUI's current directory changes depending on the dialog in use. This also compensates for the fact that prior to version 3.03 Hercules's current directory normally never changes.

## 9.5 Preferences

The Preferences dialog is where directories, file extensions, logging options etc are defined. The following subtopics describe each of the Preferences tabs in detail.

### 9.5.1 Directories

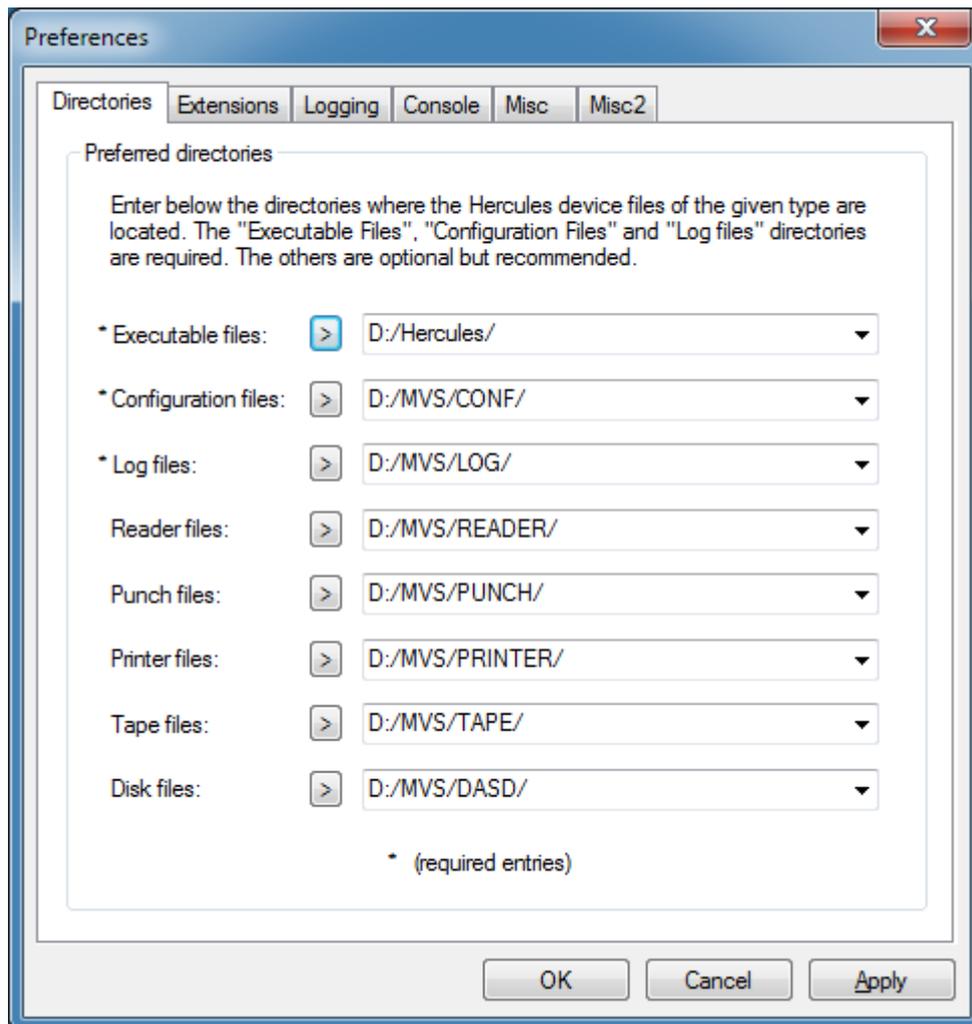The Directories tab allows you to specify the preferred directories for each file type.



**Figure 29: Preferences Directory Tab**

The path for executable files, configuration files and log files are all required, all others are optional and provided for convenience. If specified, they are used as default directories by the various device configuration dialogs.

## 9.5.2 File Extensions

The File Extensions tab allows you to specify the preferred file extensions used in the "Files of type" dropdown list in all standard 'Open' and 'Save as' dialog boxes that the Hercules Windows GUI uses.
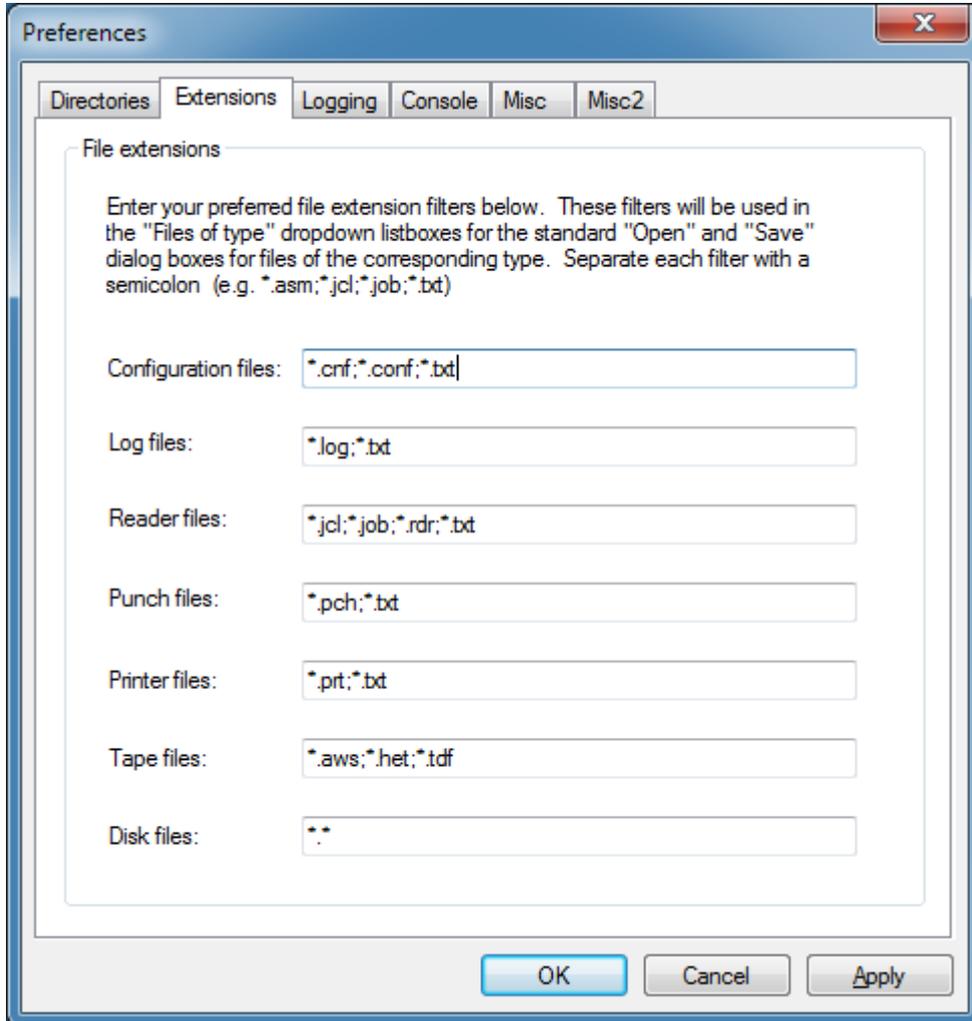


**Figure 30: Preferences Extensions Tab**

This feature allows each user to have different naming conventions for their reader, punch and disk files etc.

### 9.5.3  Logging
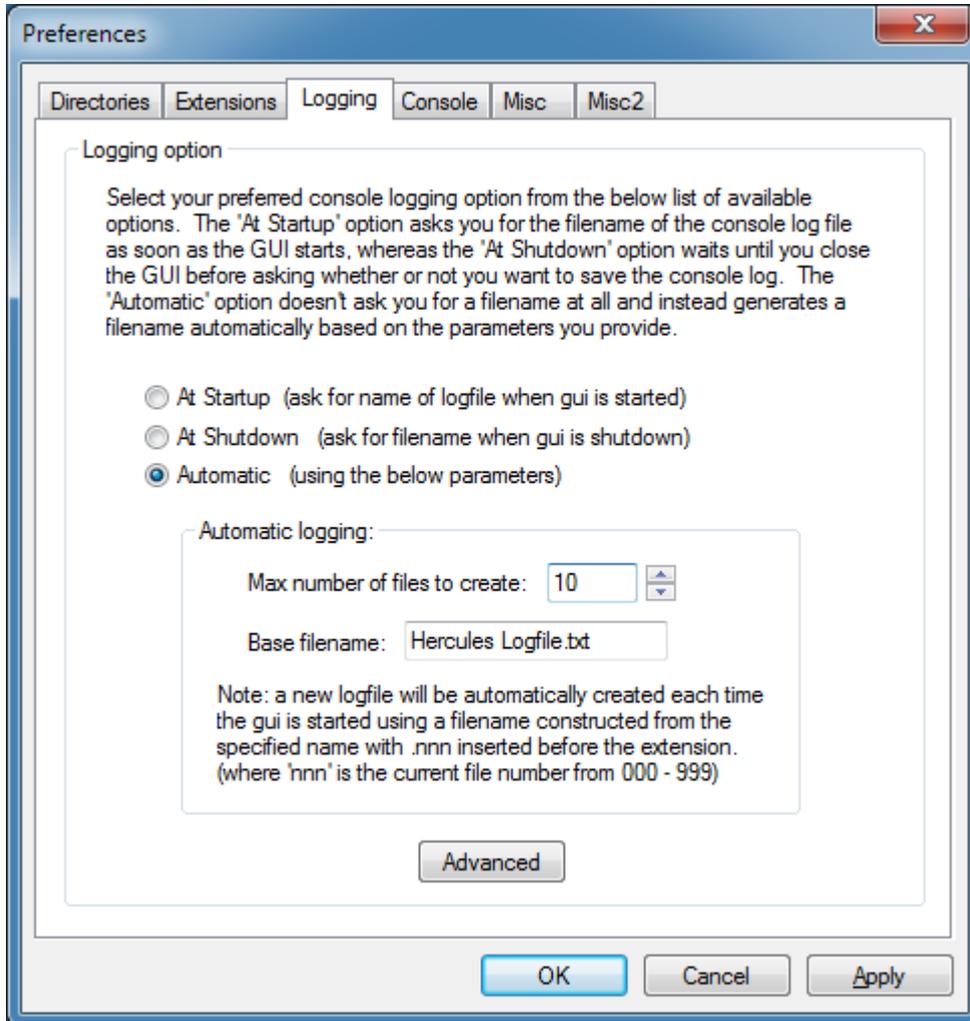
The Logging tab allows you to specify the preferred console logging options.



**Figure 31: Preferences Logging Tab**

The 'At Startup' option requests the filename of the console log file when the GUI is started.  The 'At Shutdown' option asks if the console log file should be saved or not when the GUI is about to be closed.

The 'Automatic' option does not ask for a filename at all and instead generates a filename automatically based on the parameters provided. For example, if the base filename is "Hercules Logfile.txt" the generated log filename will be "Hercules Logfile.000.txt" the first time the GUI is started, then "Hercules Logfile.001.txt" for the next start of the GUI, etc.

Starting with version 1.4.0, once log files are created by whichever method, they are written to continuously as new messages arrive. This removes the need to do a periodic "Save Messages".

As new messages are now automatically written to the log file during Hercules execution, it is recommended to use the "Advanced Logging Options" dialog to specify the maximum log file size in number of lines. This is meant to prevent the log file from filling up your hard drive.  Use the 'Advanced' button on the main Logging preferences tab to access these settings.
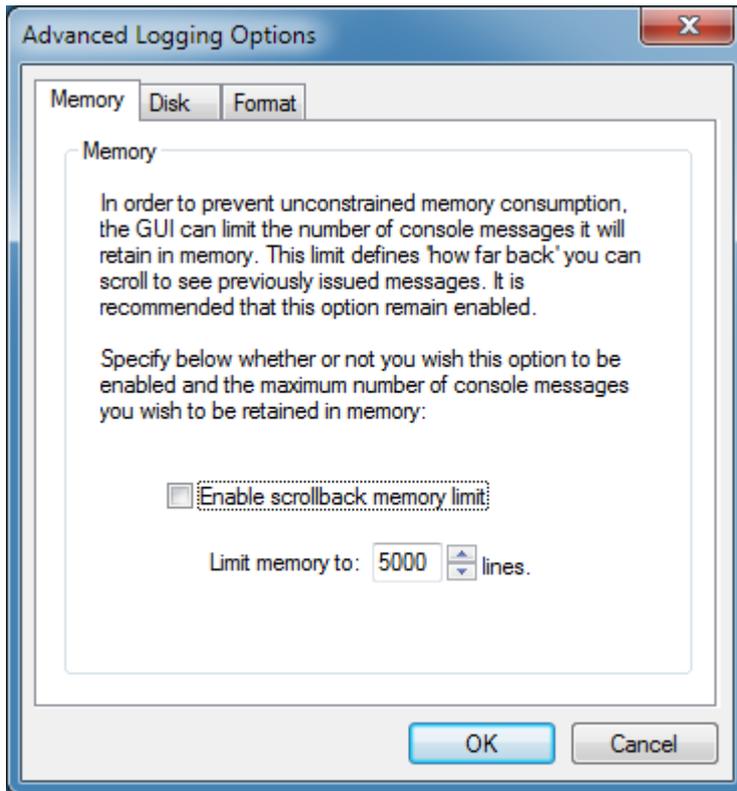
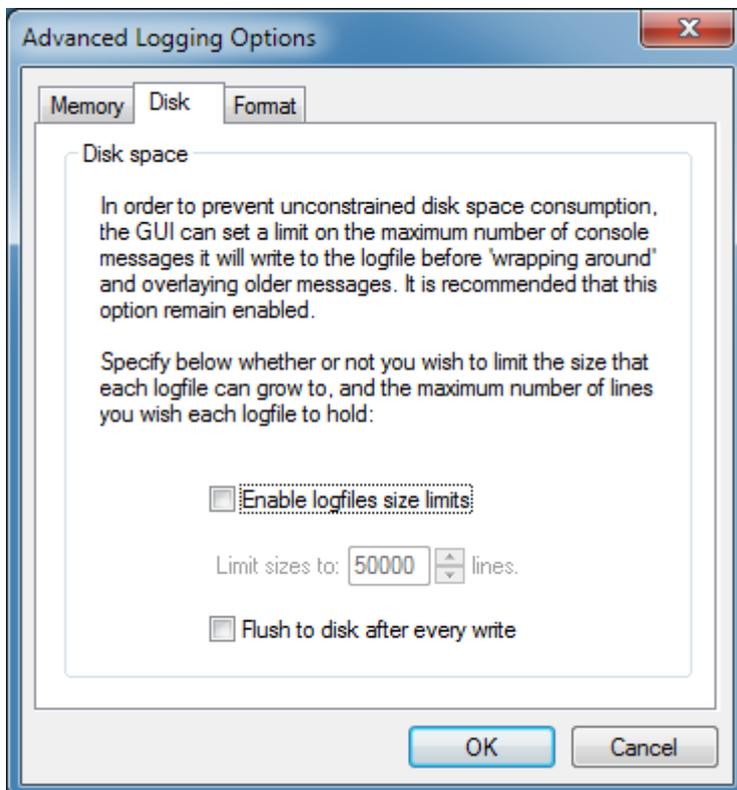**Figure 32: Advanced Logging Options Memory Tab**



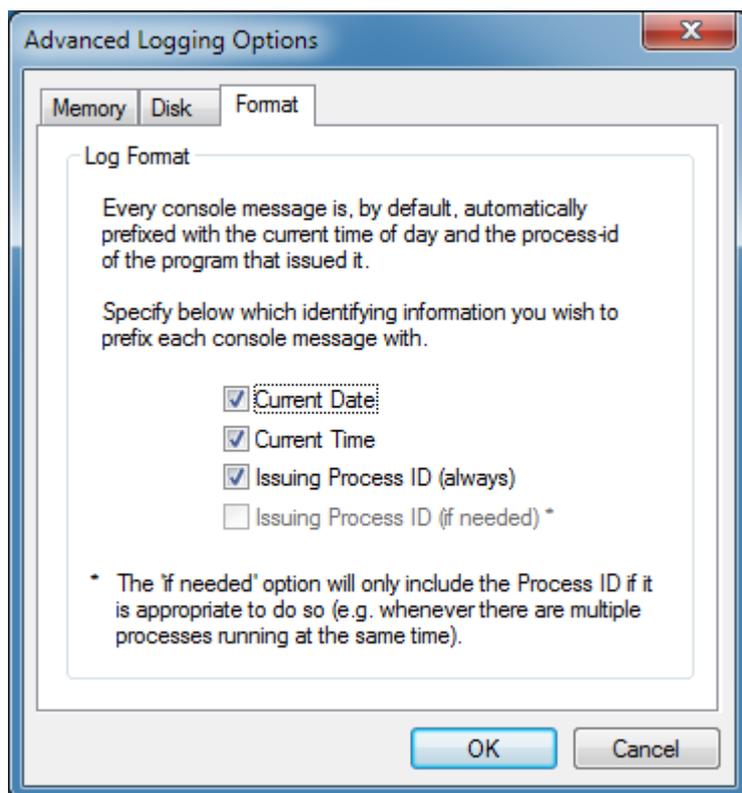**Figure 33: Advanced Logging Options Disk Tab**

**Figure 34: Advanced Logging Options Format Tab**

The log file is a wrap-around file, where the oldest messages will be overwritten by newer messages when the specified size limit has been reached. Once the specified maximum number of lines has been written to the log file, the GUI will reset its file position pointer back to the beginning of the file and begin writing new log file messages over the top of older ones.

The file is not recreated when full. When it wraps around and begins to overlay older messages, the messages near the end of the log file are still present until eventually overwritten.

Note too that the Advanced Logging Options dialog lets you specify the maximum number of messages to be retained in memory. A limit is required to prevent run-away memory consumption. This memory limit is a separate value from the disk log file limit and essentially controls how far back you can scroll the console to see older messages.

## 9.5.4 Console

The Console Tab allows it to specify the preferred console font, font colour and background colour for the Hercules console message area.
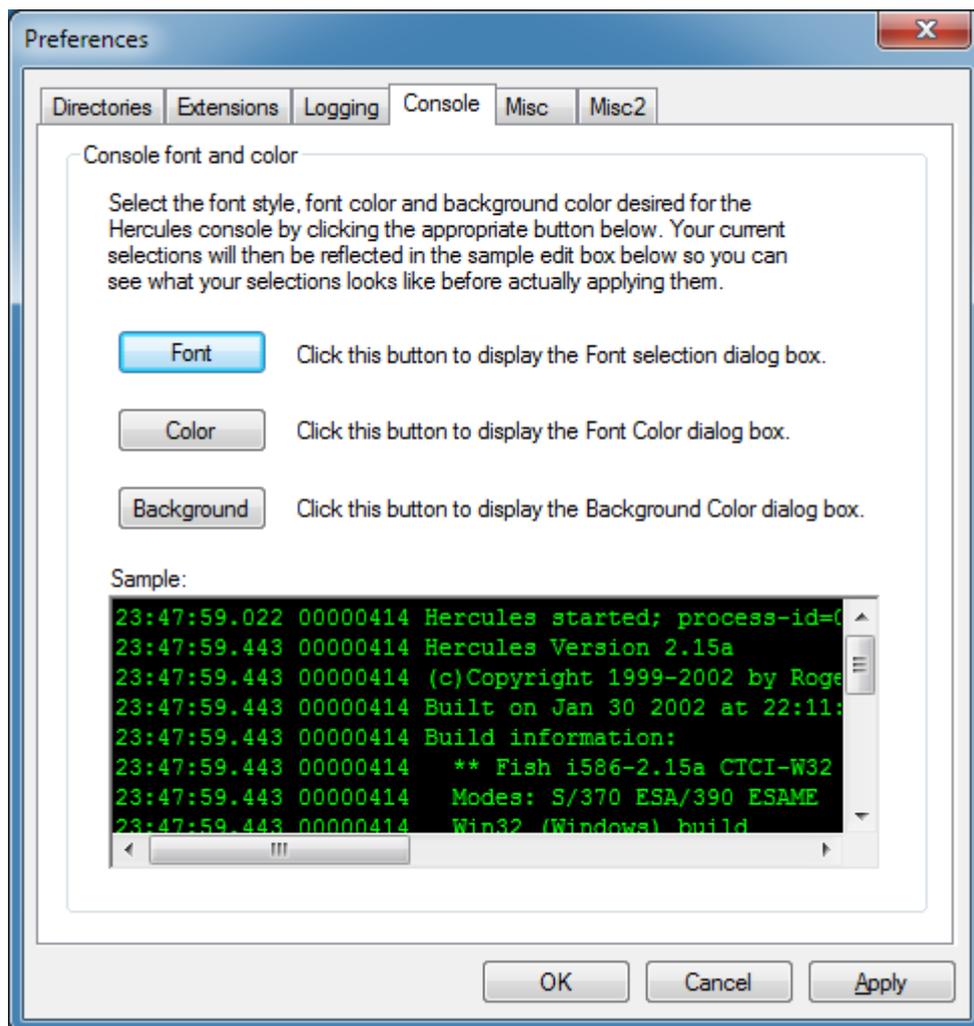
**Figure 35: Preferences Console Tab**

Simply click on the appropriate button and select the font or color. Your selection will then be reflected in the sample edit-box to let you see what the console would actually look like before your changes are applied.

## 9.5.5  Misc

The Misc Tab allows you to specify various miscellaneous preferences, such as how the GUI should react to your pressing the "Power Off" button.
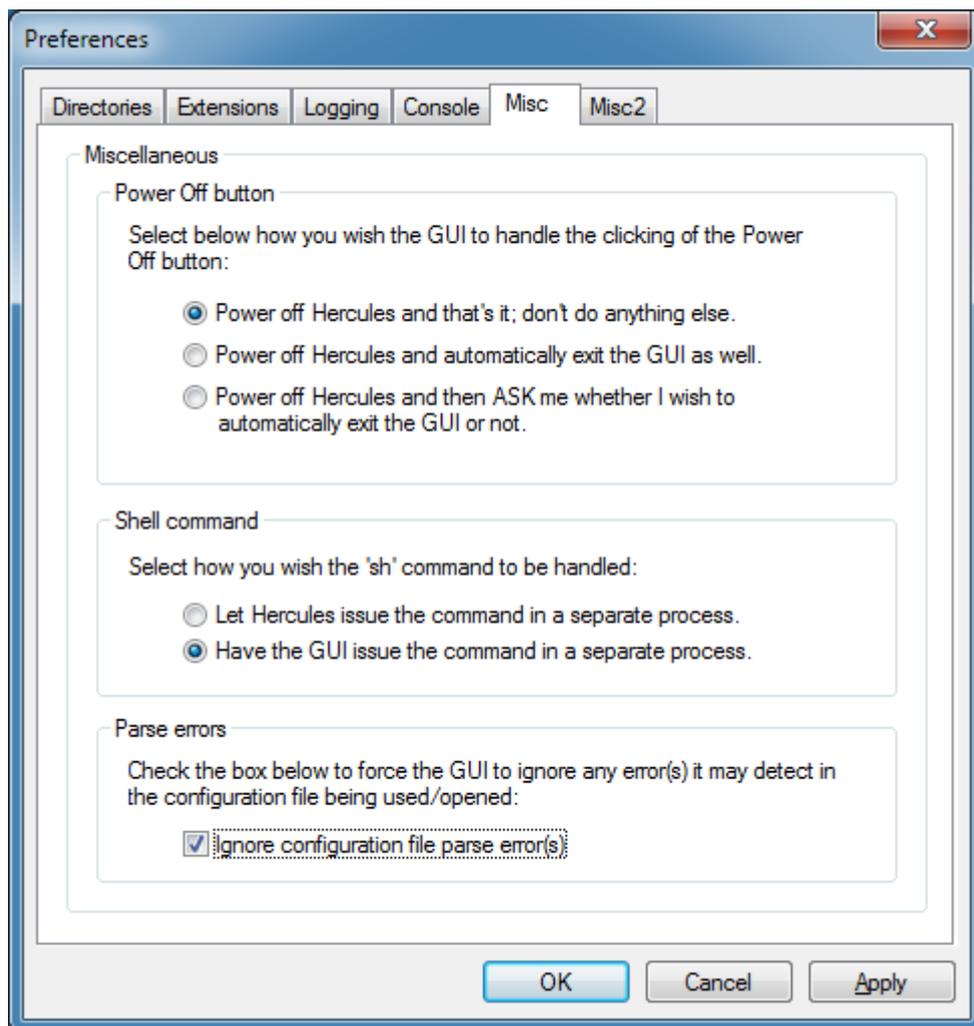
**Figure 36: Preferences Misc Tab**

When shell command support was added to Hercules for the Windows environment a bug was revealed in the manner that Cygwin processed the 'fork' command. Although a workaround was developed the GUI was also modified to be able to issue the shell commands directly instead.

In order for the GUI to be able to issue shell commands instead of Hercules itself a new program called 'conspawn.exe' was developed. The GUI passes the shell command to conspawn for execution. The conspawn program is included as part of the GUI package and must reside in the same directory as the other Hercules executables.

The Ignore Parse Errors option was created to allow you to open and use a Hercules control file that the GUI would otherwise fail to parse properly for whatever reason.

When a control file is opened the GUI parses the statements and saves the information in an internal control area. This information is used later to complete various fields in the dialogs such as the device configuration dialog. If the GUI cannot properly parse a given control file statement it throws a parse error and prevents you from opening or using what it considers to be a bad control file. This option can be used to bypass this error, for example when using a device statement containing a new parameter that Hercules understands but the GUI may not yet support.

The "Ignore Parse Errors" option instructs the GUI to ignore the parse error and open and use the control file anyway. When this option is set the GUI ignores all parse errors and will always successfully open whatever control file you instruct it to. As the GUI can update the Hercules control file, it is highly recom-

mended that you leave this option disabled and enable it only when needed to avoid accidentally damaging a file that may not contain valid Hercules control specifications.

If you do need to enable this option, remember to disable it when no longer needed as once enabled it will stay that way until you purposely disable it. Like all preference options it is persistent across executions of the GUI.

## 9.5.6  Misc2

The Misc2 Tab provides additional options that would not fit on the original Misc options tab.
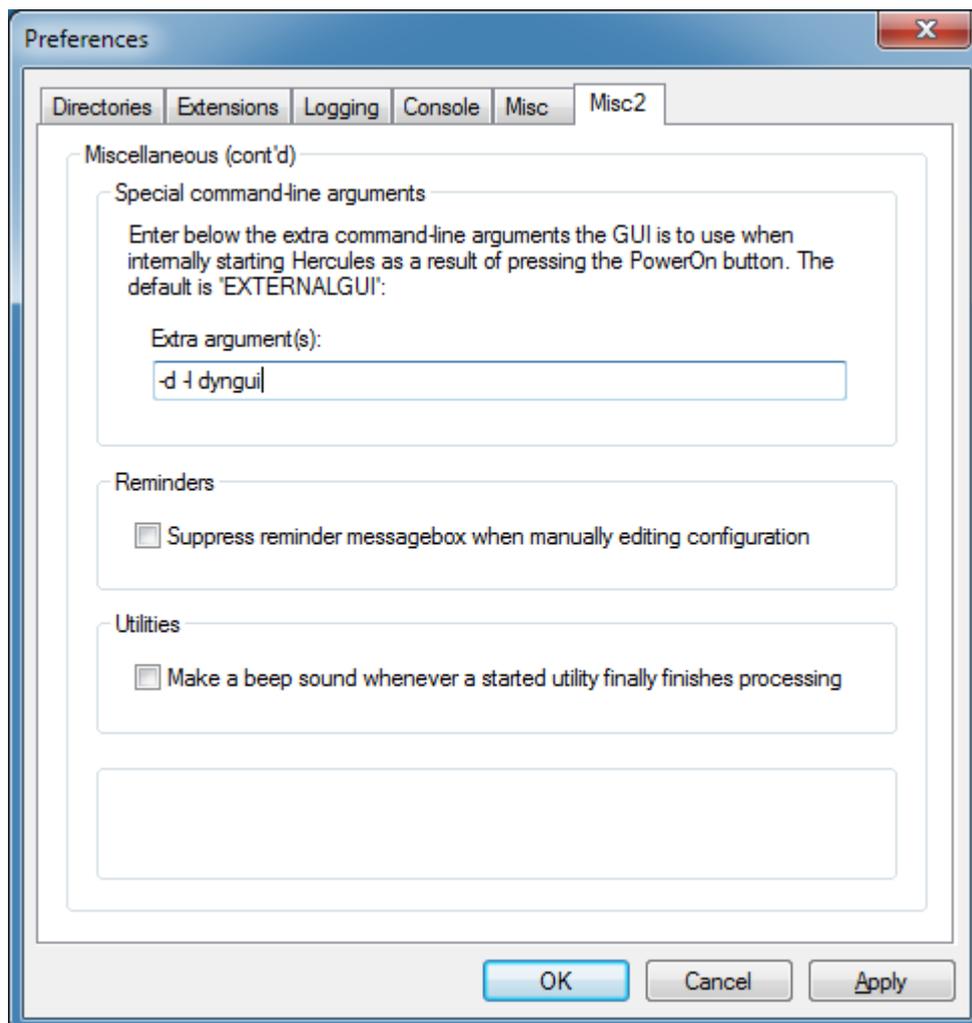


**Figure 37: Preferences Misc2 Tab**

Since version 3.0 of Hercules support has been added for command-line options beyond the existing -f (control file) option. Here you can enter any of the new command-line options you may wish to have issued whenever the GUI starts up ("Powers On") the Hercules emulator.

Note that you must not specify the -f option here. The GUI constructs the -f option automatically based on the configuration file previously defined in the Preferences dialog. You may enter any other command-line options in addition to the required ones - see the important note immediately below.

You must specify here either "EXTERNALGUI" (without quotes) or – starting with Hercules Version 3.0 – a new command line option "-d -l dyngui". This option must not be left blank.  If you accidentally leave this option blank it is possible that both Hercules and the GUI will not operate at all and highly likely that they will not operate as expected.

This is an advanced option and should not be modified by end users. Please only change this if instructed to by Hercules Technical Support or Hercules developers.

## 9.6  System Configuration

When a Hercules control file is opened by the GUI it is parsed and then a "System Configuration" dialog is displayed showing the various configuration settings.

Since HercGUI version 1.9.5 the System Configuration dialog information is displayed in three separate property pages: the Architecture page, the O/S Tailor settings page and the Other / Misc page.

The Identification section simply displays the full pathname of the configuration file that was opened and provides an input field for a text description to be associated with this particular configuration file. This is in case you have multiple system configurations where it is useful to have some indication that you are modifying the correct one. The description you enter here is saved as a comment at the beginning of the configuration file.

### 9.6.1  Architecture Settings

This page of the System Configuration dialog is where you define various hardware and architectural settings, e.g. how many emulated CPUs, how much emulated main storage etc.

The Architecture radio buttons allow you to define the default architectural mode your emulated CPUs will be initially started in. For emulated z/Architecture machines the actual IPL takes place in ESA/390 mode and it is the responsibility of the IPL'ed operating system to switch the CPU(s) into z/Architecture mode when it is ready to use this mode.  It does this using the SIGP instruction.

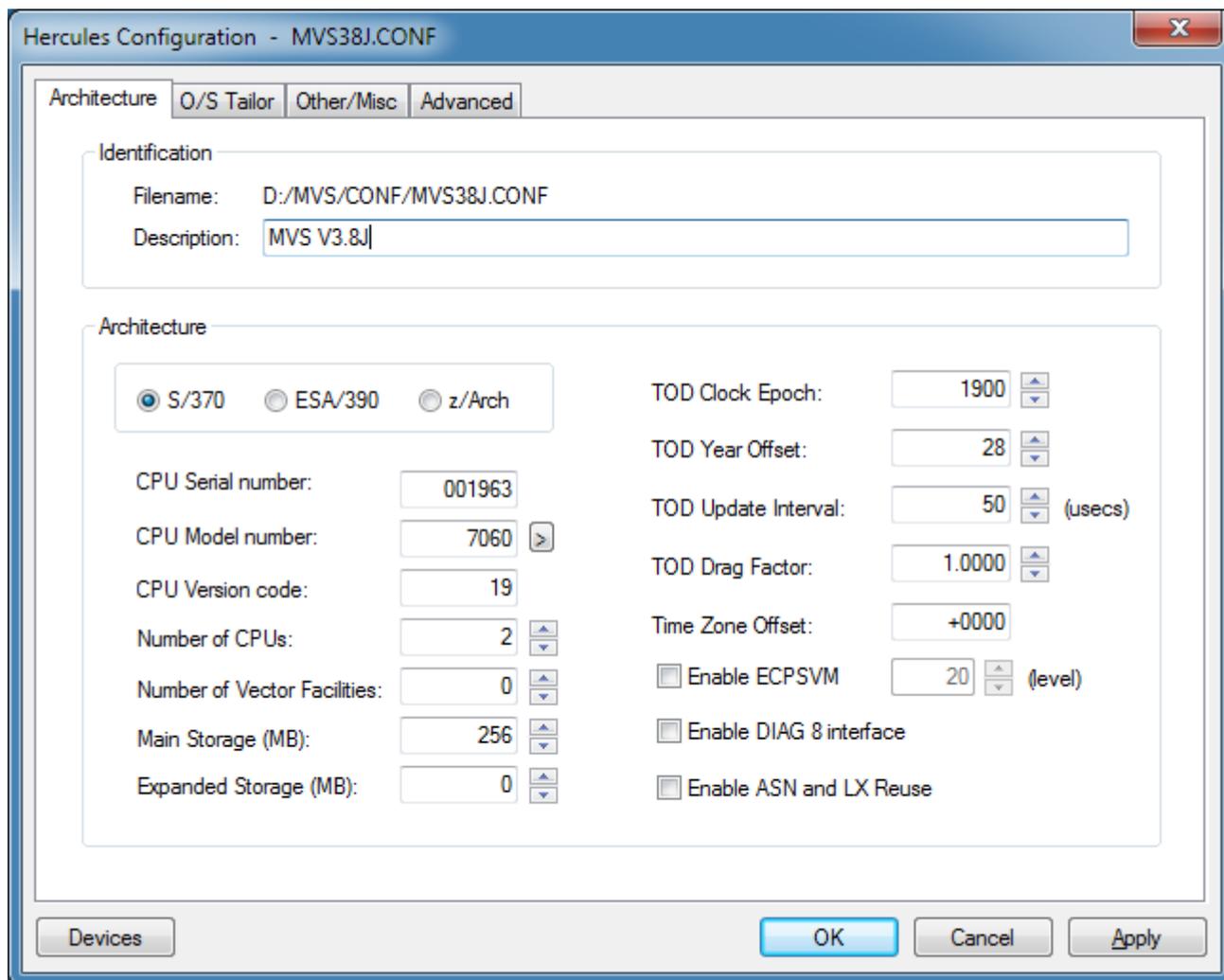The following figure shows the Architecture Settings tab.

**Figure 38: Architecture Settings Tab**

In the main System Configuration dialog, if the GUI finds a specially formatted file called "cpu-types.txt" in your preferred configuration files directory, a small '>' button will appear next to the "CPU Model number" edit-box. This file may contain a list of the various CPU model numbers, their names and their corresponding STIDP (Store CPU ID instruction) values.

When the System Configuration dialog is initialized, if the GUI finds this file, it will parse the entries and use them to construct a drop-down box that will be displayed when you click the '>' button. This allows you to easily select the desired CPU Model you wish your virtual mainframe to be reported as.

Neither the GUI nor the Hercules Emulator itself makes any attempt to try and emulate all aspects or features of a given CPU model. This CPU model number simply specifies what value to use in the STIDP (Store CPU ID) instruction.

When your CPU Model is selected in this way the GUI automatically fills in the "CPU Model number" edit-box field with the corresponding value it finds in the "cpu-types" file. You have the option of manually overriding this value. A sample "cpu-types" text file is included with the distribution of the Hercules GUI.

## 9.6.2 O/S Tailor Settings

The O/S Tailor settings page is where you can establish certain settings related to the type of guest operating system you intend to actually run on your virtual mainframe. The purpose of the O/S Tailor radio buttons is to limit the amount of Hercules generated message traffic by selectively suppressing certain program check and trace type messages which are considered normal for the specified operating system.
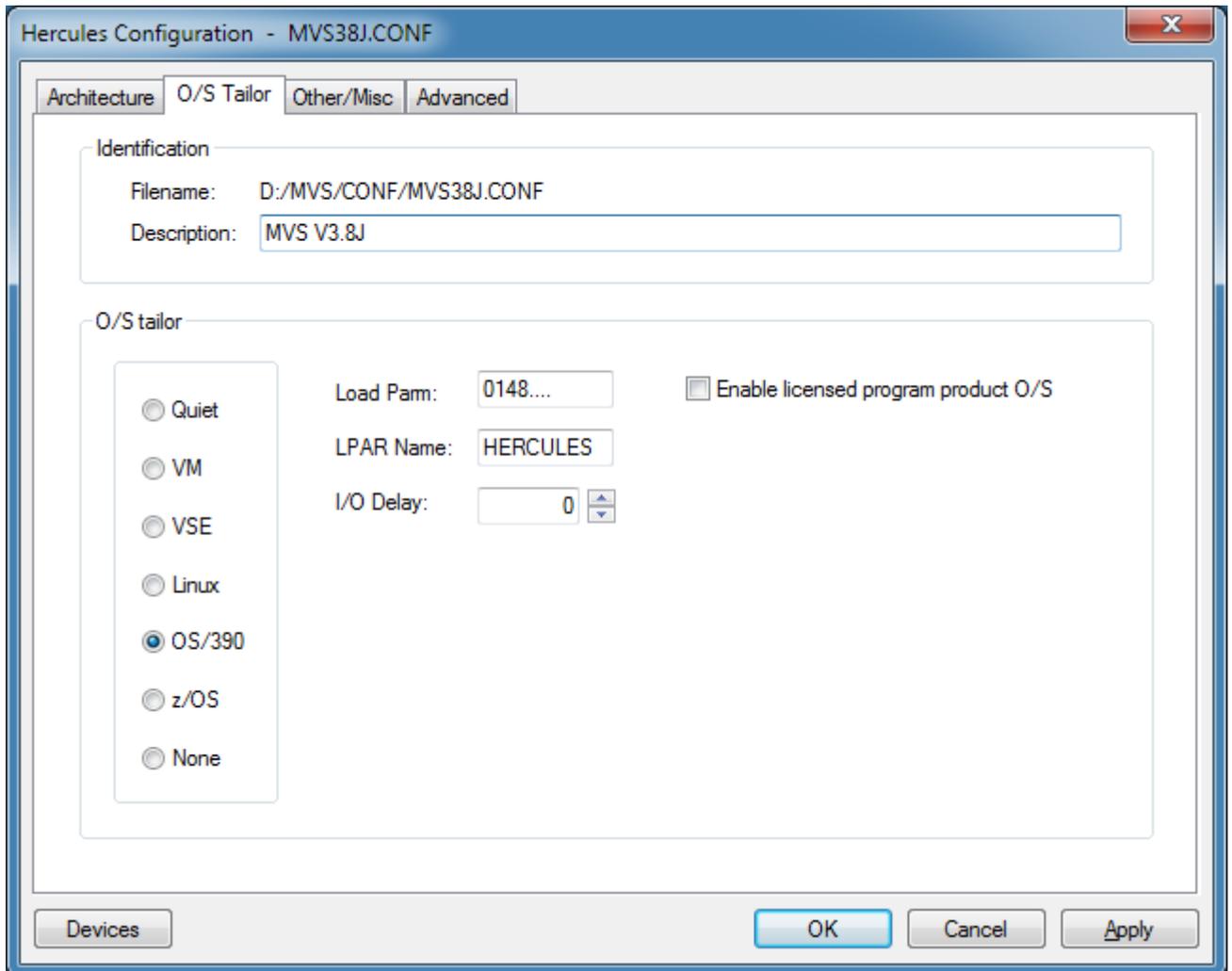


**Figure 39: O/S Tailor Settings Tab**

When the "Enable licensed program product O/S" option is specified for a given control file any attempt to power on Hercules using that control file will result in a dialog box being displayed that asks you to verify your true intentions.
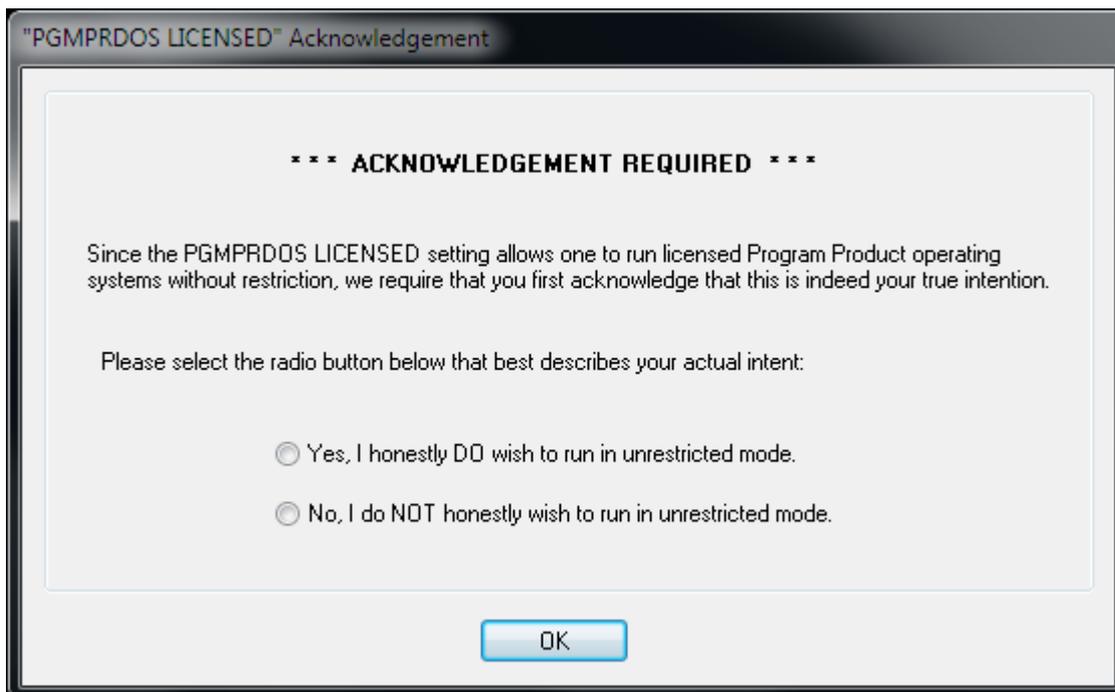
**Figure 40: PGMPRDOS LICENSED Acknowledgment**

As soon as this dialog is displayed a 10-minute timer is started. If you respond before the timer expires your response is accepted as-is. If your response is "No, I do NOT honestly wish to run in unrestricted mode" then your virtual mainframe will not be powered on. If your response is "Yes" - whether explicit or presumed (see next paragraph) - then Hercules will be powered on.

Please note that if you fail to respond within the 10-minute time limit your response is presumed to be "Yes, I honestly DO wish to run in unrestricted mode". If you do not wish to wait the entire 10 minutes then you will have to respond to the dialog manually yourself. There is no way to disable or override this feature.

This dialog is displayed each time you attempt to power on Hercules during a given Hercules GUI session when using a configuration file with the "Enable licensed program product O/S" option checked. Although you only have to respond to this message once during a single session if you continue to use the same configuration file. If you switch to a different configuration file and then return later to the first one (or exit the GUI entirely and start it again), you will be asked once again to confirm your intent.

## 9.6.3  Other / Misc Settings

The "Other / Misc" window allows you to define more system configuration values, mostly related the internal functioning of the Hercules emulator. Please refer to the documentation for the Hercules emulator itself for more information regarding the various values that may be specified here.
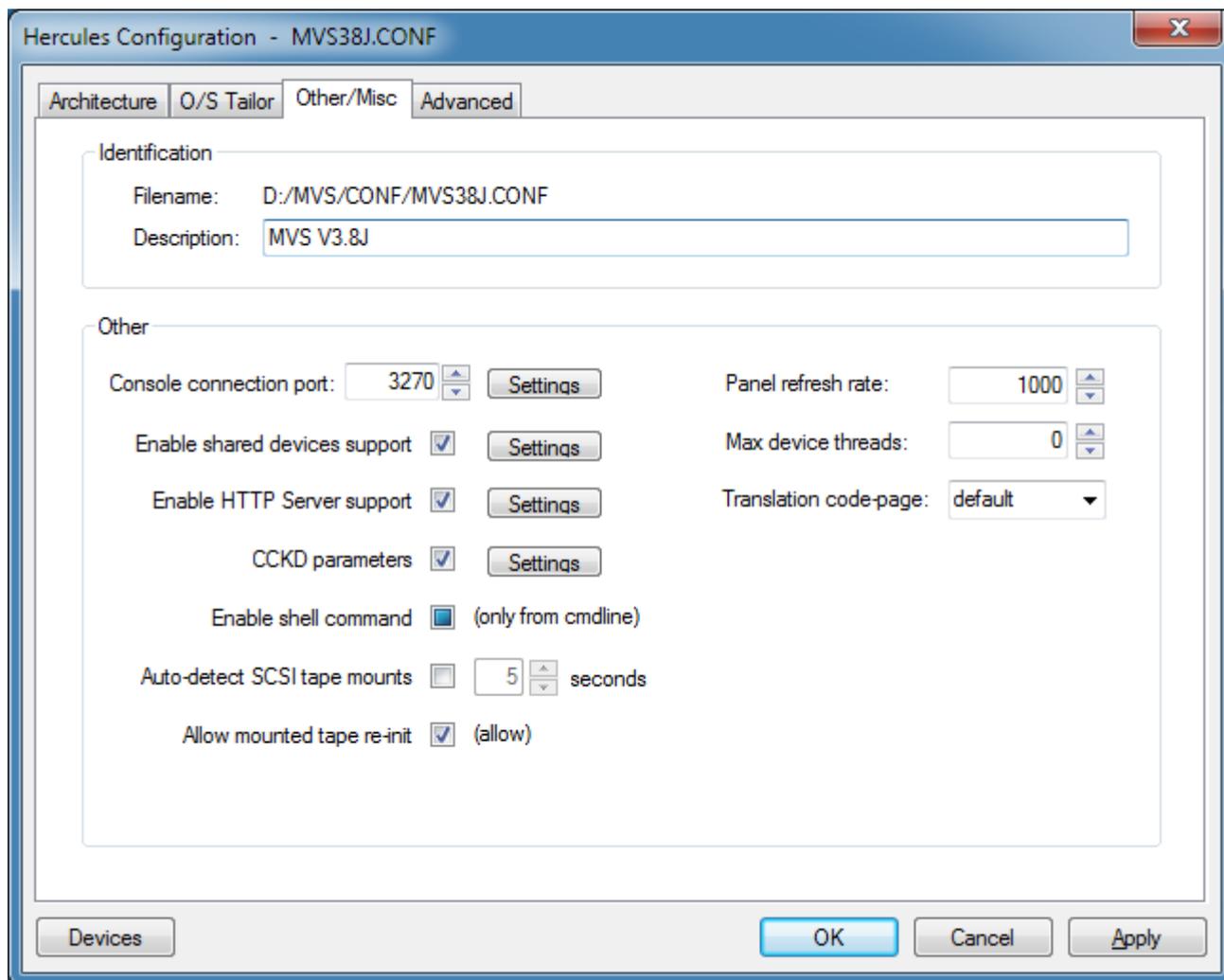
**Figure 41: Other / Misc Tab**

Within this tab several other settings can be specified. If the "Enable shared devices support" checkbox is marked, then the settings window can be opened which allows to specify the port for the Shared Devices Server.
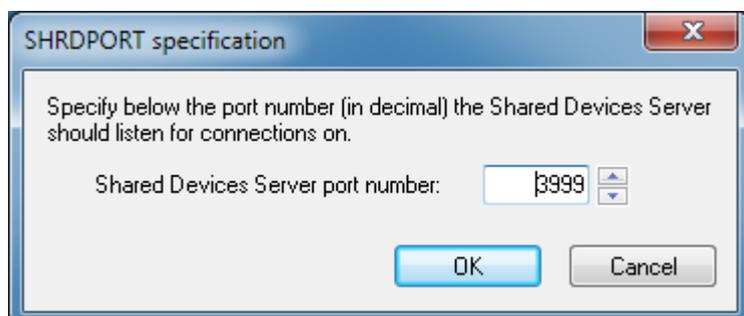


**Figure 42: SHRDPORT specification**

Hercules's HTTP Server support allows you to control Hercules via any standards compliant web browser. This dialog allows you to define the HTTP Server parameters that Hercules is to use.
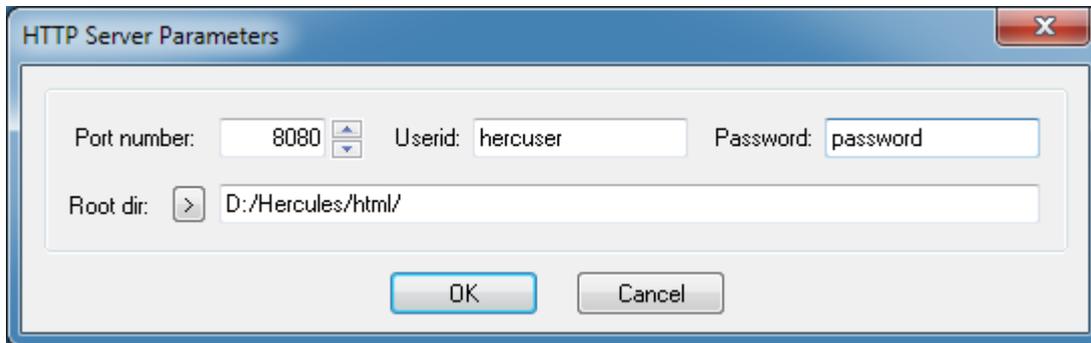


**Figure 43: HTTP Server Parameters**

Enter the port number for Hercules's HTTP Server to listen on and if desired specify the authentication criteria needed to connect to the server. You can also enter the root directory from which the web pages will be served.

If you enter a userid for authentication then you must also enter a password. If no userid and password are entered then anyone with a browser that is able to connect to your Windows host system will be able to control your Hercules system via the HTTP Server interface. The password you enter is not encrypted in any way and is stored in your Hercules control file, as well as passed through the network, in unencrypted plain text format. You should therefore take whatever steps are required to secure Hercules control file(s) that contain HTTP Server passwords.

Since Hercules version 2.17 the behavior Compressed CKD DASD (CCKD) functionality is controlled via the setting of certain global parameters.  CCKD functionality is no longer adjustable on an individual device-by-device basis.
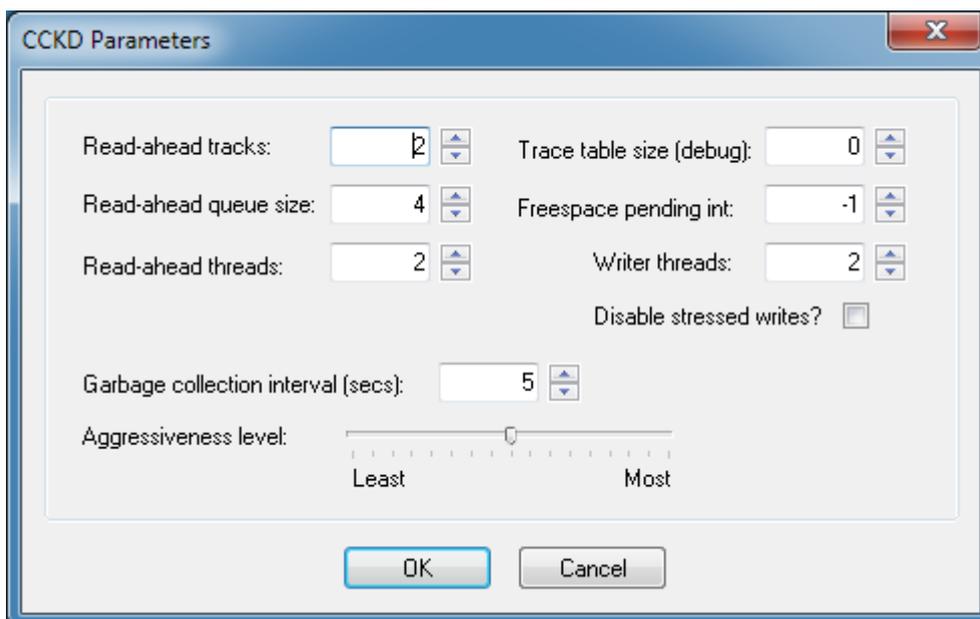


**Figure 44: CCKD Parameters**

### 9.6.4  Advanced Settings

The Advanced configuration page is where settings for features that are intended only for more advanced users may be made. If you have a custom dynamic module (DLL) you wish Hercules to use or wish to modify Hercules's default priority settings you would do that here.

Please see the "Hercules User Reference Guide" for more details on the options presented here.
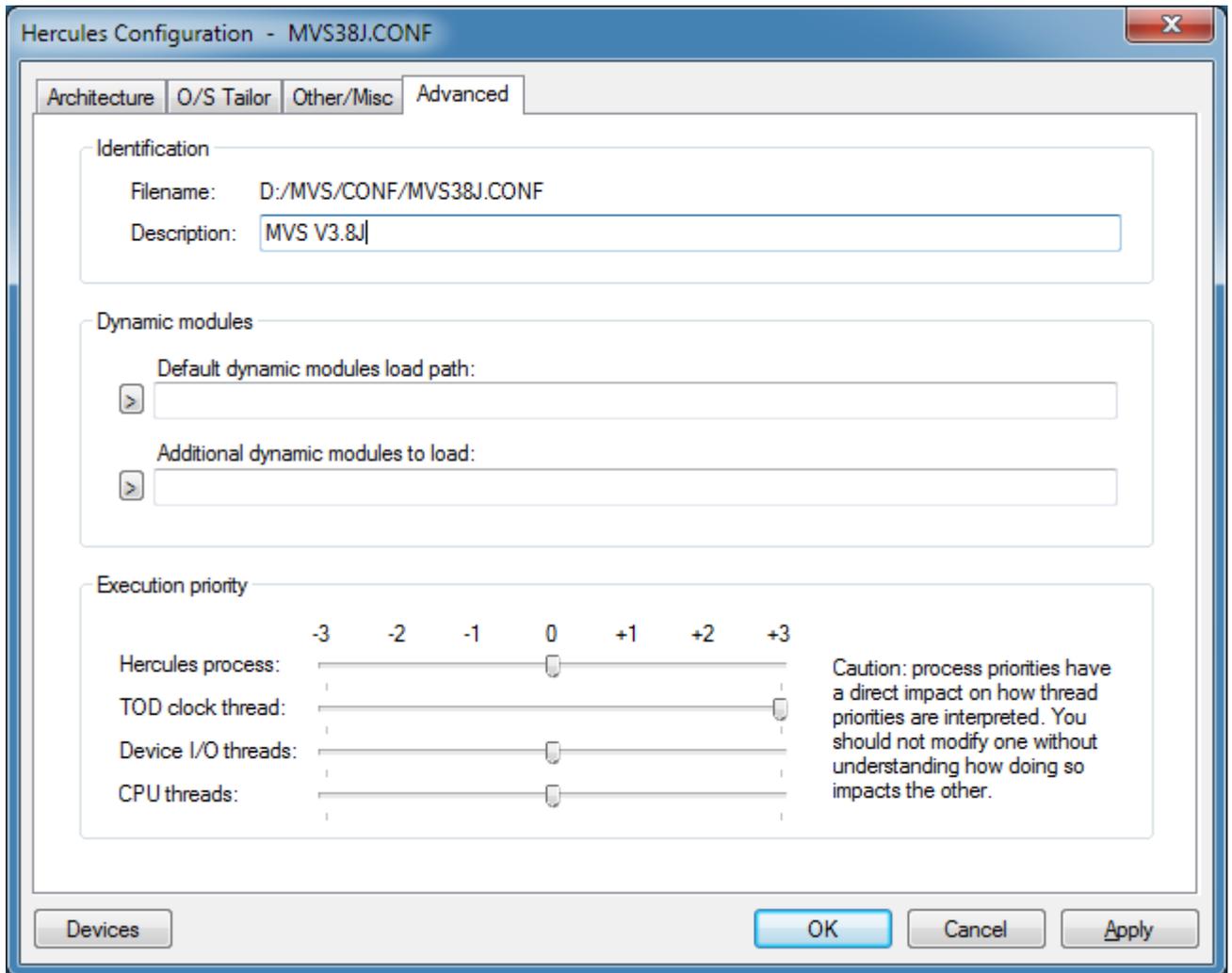


**Figure 45: Advanced Tab**

## 9.7  Device Settings

Clicking the "Devices" button from the System Configuration dialog or selecting "Modify Devices" from the File menu will take you to the Device Configuration dialog. From here you can add, delete or modify the devices in the current configuration.

This particular dialog is resizable as device configuration statements can be quite long.
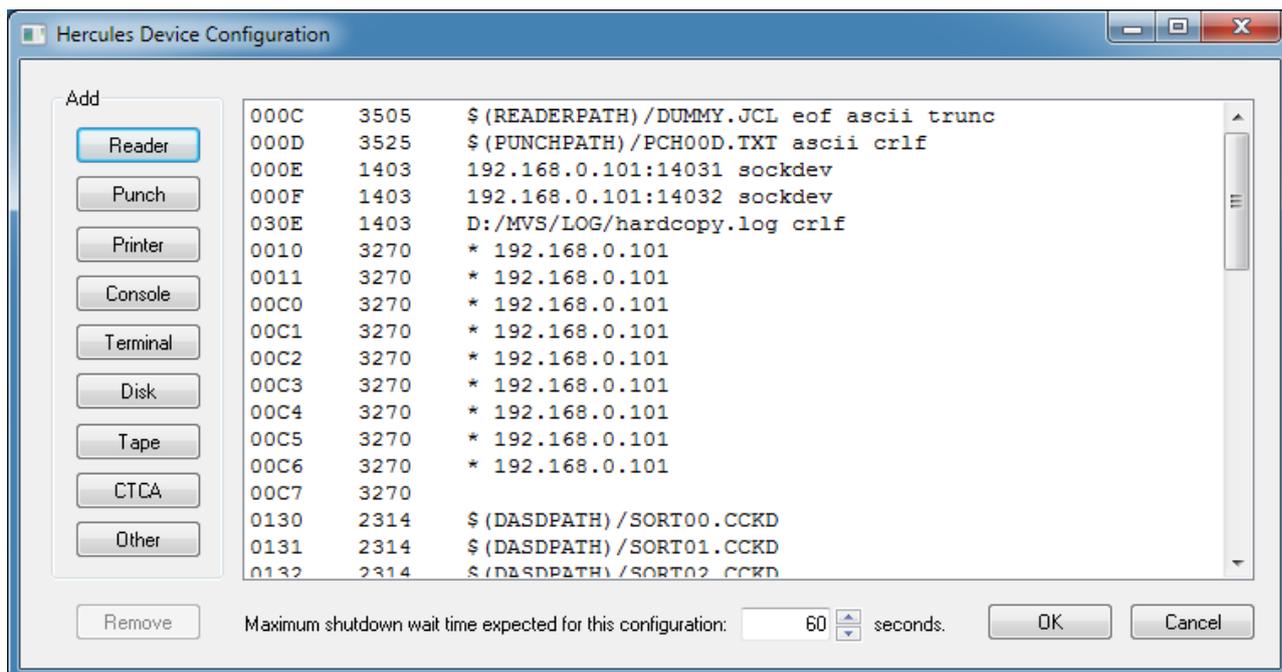
**Figure 46: Device Configuration**

The *"Maximum shutdown wait time expected for this configuration"* setting defines a time limit to the Hercules GUI. This limit is the amount of time the GUI is to expect between when the 'quit' command is issued (or the "Power Off" button is pressed) and when Hercules finally finishes exiting after completing its shutdown sequence.

When you use compressed disks (CCKD) Hercules needs time to write-back cached copies of track images and adjust the free space for each disk before it can safely exit. If the expected wait time is exceeded the GUI issues a warning asking whether to continue waiting or forcibly terminate the Hercules Emulator process. If many compressed disks are frequently updated it can take over a minute to write all cached data to disk.

It is safe to specify a value for this setting that is high enough for all likely cases in your environment. The Hercules GUI will terminate as soon as Hercules itself ends regardless of the wait time setting.

Right clicking on a device statement presents a context menu from which you can select 'Edit' or 'Properties'. Selecting 'Properties' presents the "Reinitialize Device" dialog, also displayed by double-clicking the device statement.

When 'Edit' is selected from the right-click context menu you are presented with a simple device statement edit dialog. From here you can directly modify raw device statements.
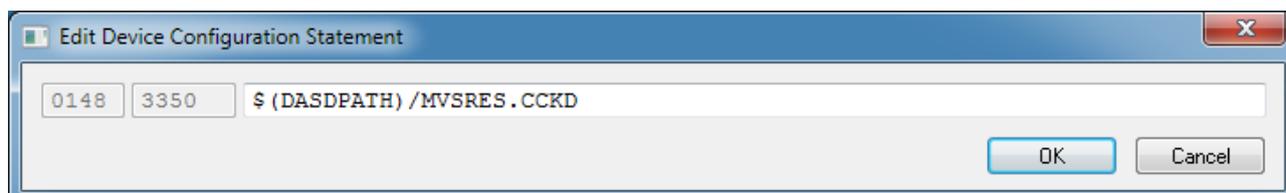


**Figure 47: Edit Device Configuration Statement**

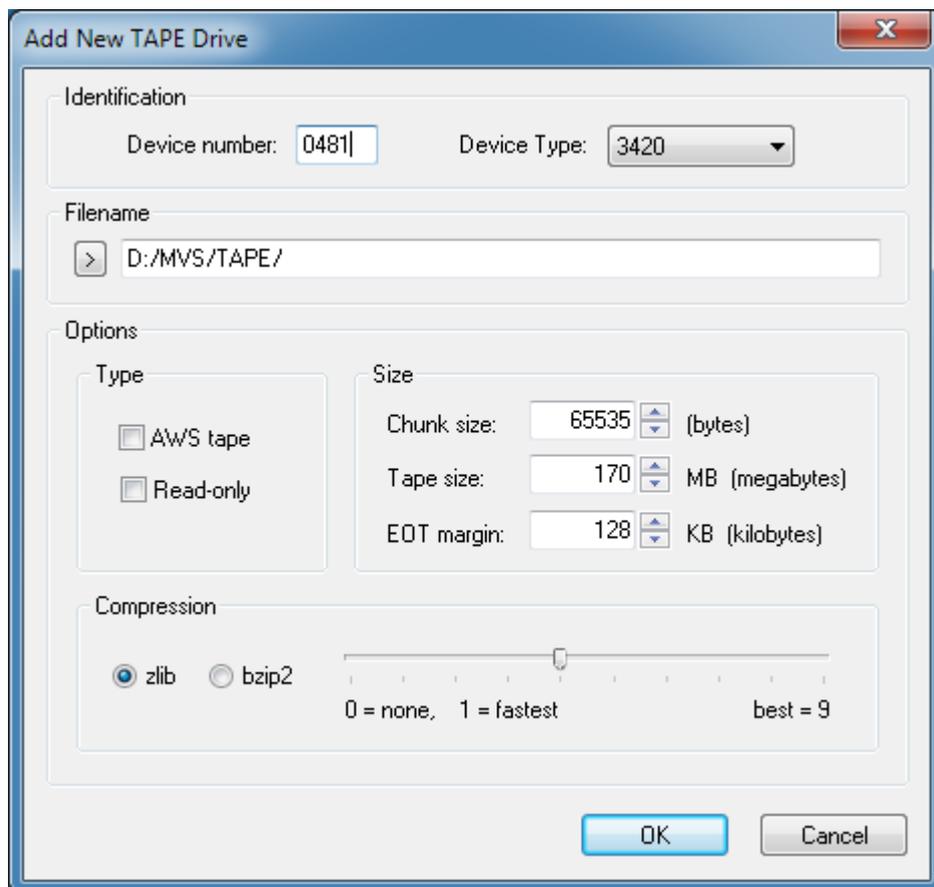To add a new device, click on one of the 'Add' buttons:



**Figure 48: Add New Device**

To delete a device select the desired device first to highlight it and then click on the 'Remove' button. To modify (reinitialize) an existing device double-click on the entry for the desired device:

**Figure 49: Reinitialize Device**

Note: If you used any of the new CCKD parameters on the main System Configuration dialog, some controls in the above dialog will not be displayed. This occurs because the new System Configuration CCKD parameters modify CCKD functionality on a system-wide basis and remove the ability to specify these parameters on per-device basis.

## 9.8   Display / Alter Memory

The "Display / Alter Memory" item in the command menu allows you to display or modify absolute main storage.

It is very important that you keep in mind that when you alter absolute main storage via this dialog then neither the storage keys nor the CPU instruction and data caches are updated in the Hercules emulator itself. Instead the memory of the Hercules emulated operating system is directly modified without the Hercules Emulator knowing of this.

Please use this feature with caution. It is designed for examining and searching main storage for emulator debugging purposes and not as a safe means of modifying hosted operating system storage.

The following figure shows the "Display / Alter Memory" dialog.
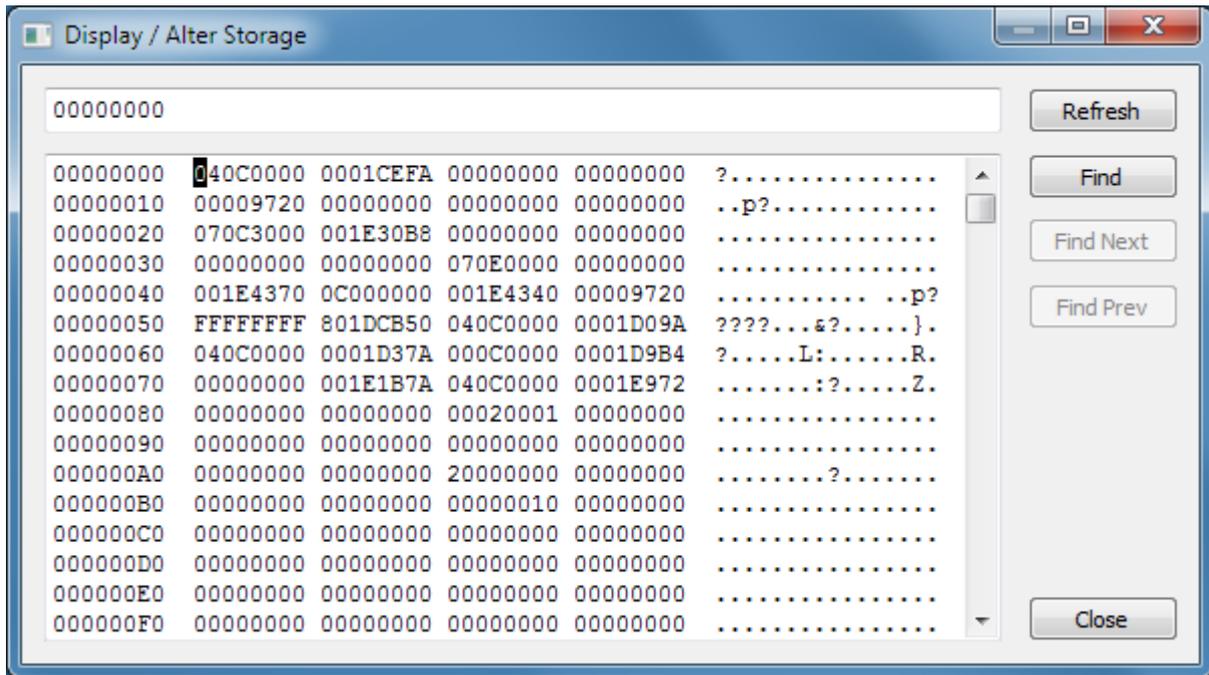
**Figure 50: Display / Alter Memory Dialog**

If you need to modify real or virtual storage it is highly recommended that you use the Hercules Emulator 'r' and 'v' panel commands. These ensure that the hardware emulator is aware of any changes.

## 9.9 Load Card Reader, Load Tape, Unload Tape

These menu items provide a quick and easy way to do just as their descriptions suggest. The 'Load Reader' command is provides an interface to submit jobs to the system. It displays the "Reinitialize Device" dialog for the card reader. From here you can use standard Windows 'Open File' dialogs to browse for a file that you want to submit. The selected file will be loaded into the card reader. Clicking OK issues the appropriate Hercules devinit panel command.
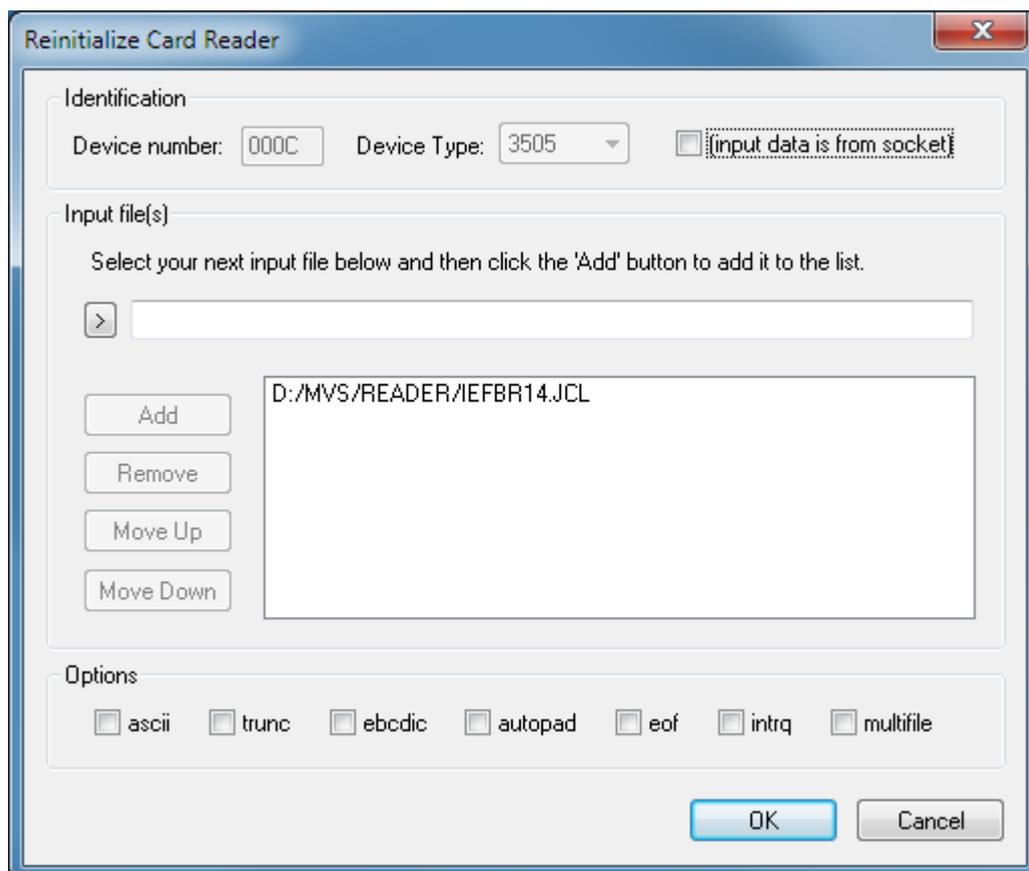
**Figure 51: Reinitialize Card Reader Dialog**

The "input data is from socket" option is found to the right of the device type field. It allows you to tell Hercules to obtain card reader input from a specified socket instead of a disk file as by default. This allows you to submit card decks remotely using a simple utility that connects to the specified socket and writes card images directly to Hercules. Recent releases of the GUI provide a DOS program "HercRdr" to support this capability. For more information on the HercRdr utility and the 'sockdev' option refer to the Hercules User Reference Guide.

## 9.10 Device List Bar

The "Device List" bar, similarly to its non-GUI console mode counterpart, lists the devices in the current configuration and their status.
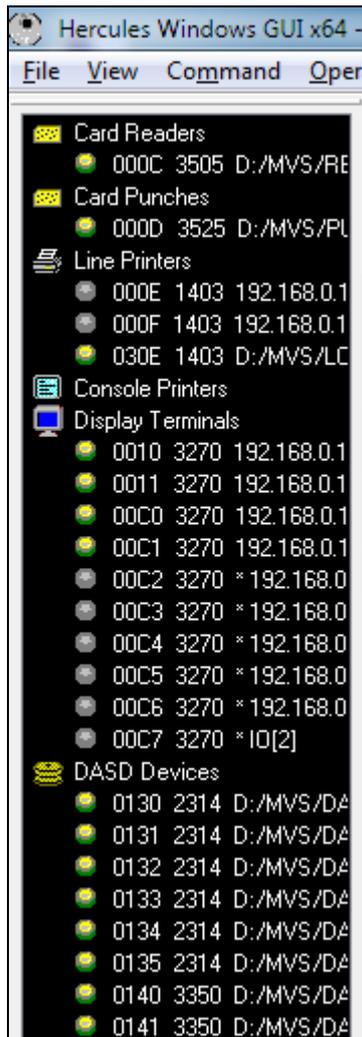
**Figure 52: Device List Bar**

A grey diode indicates the device is offline and not open. A green diode indicates it is online or open. The green diode changes to yellow whenever the device is busy and changes to red when there is an interrupt pending for the device.

As the hosted system runs and performs I/O to the devices in your configuration you will see the diodes change between yellow, red and green. This shows that there is I/O activity taking place on the device.

Devices are displayed in a tree-list with a branch for each class of device. Right-clicking the Devices within a branch presents a context sensitive menu. You can also right-click each branch.

To add a new disk drive to your configuration right-click on the "DASD Devices" branch and select 'Add device' from the menu that appears. To delete ("detach"), rename ("define"), reinitialize ("devinit") or present an attention interrupt ("i") for a particular device, right-click the device and select the appropriate option from the context menu presented.

## 9.11 Utilities Menu

All of the Hercules utility programs can be run by completing the appropriate dialog. Both Hercules and these utilities run as separate processes, so it is possible to run more than one utility at the same time as Hercules images.

A progress dialog is displayed as each utility runs, all messages generated by the utility are displayed on the GUI console just as Hercules messages are. Each message is prefixed with its process ID to differentiate between utility program messages and a timestamp.

The following figures show an example of the DASDINIT utility window.
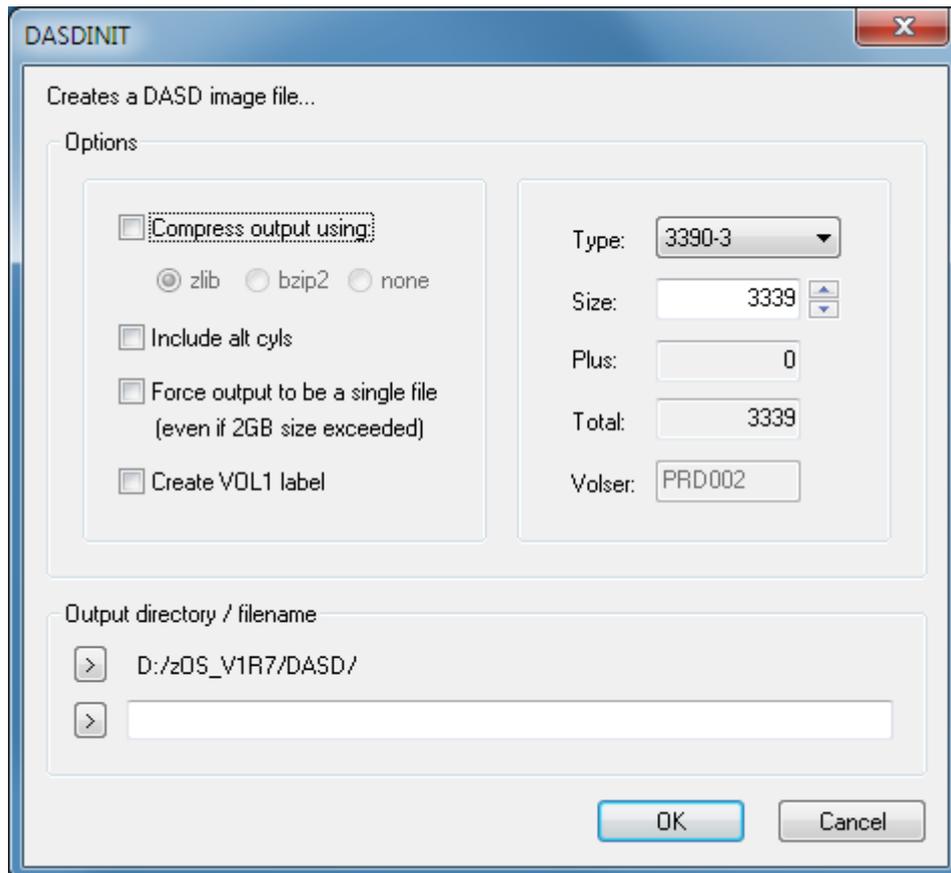


**Figure 53: DASDINIT Utility Window**

## 9.12 Registry Tweaks

Some of Hercules minimum, maximum and default values used by the GUI are stored in the windows registry. The following table shows the valid values.

Apart from the "MinTODDrag" and "MaxTODDrag" which are string values, all of the values below are DWORD values and are stored in the "Limits" branch of the main Hercules Windows GUI registry key:

"HKEY_CURRENT_USER/Software/Software Development Laboratories/Hercules/Limits"

Note that changing the values of these registry entries will not necessarily change the actual function of the Hercules Emulator. If the GUI accepts the new value this does not necessarily mean that Hercules itself will accept the value.

| Name | Default | Description |
|---|---|---|
| MaxCPUs | 32 | Maximum allowable number of central processors |
| MaxVectors | 4 | Maximum allowable number of vector facilities |
| MaxMainMem | 2048 | Maximum allowable amount of central storage (MB) |
| MaxExpandedMem | 1024 | Maximum allowable amount of expanded storage (MB) |
| MinEpoch | 1801 | Minimum allowable clock epoch year |
| MaxEpoch | 2099 | Maximum allowable clock epoch year |
| MinPanRate | 10 | Minimum allowable panel refresh rate (milliseconds) |
| MaxPanRate | 5000 | Maximum allowable panel refresh rate (milliseconds) |
| MaxTapeSizeMB | 2048 | Maximum allowable emulated tape size (MB) |
| MaxEOTMarginKB | 2048 | Maximum allowable emulated tape 'end-of-tape warning area' margin-size (KB) |
| MaxECPSVMLevel | 99 | Maximum allowable ECPSVM value |
| DefECPSVMLevel | 20 | Default ECPSVM value |
| MinTODDrag | 0.0001 | Minimum allowable TOD clock drag factor |
| MaxTODDrag | 10000 | Maximum allowable TOD clock drag factor |
| MinCCKDgcparm | -8 | Minimum allowable CCKD parameters garbage- collection aggressiveness level |
| MaxCCKDgcparm | +8 | Maximum allowable CCKD parameters garbage- collection aggressiveness level |
| DefCCKDgcparm | 0 | Default CCKD parameters garbage- collection aggressiveness level |
| MaxCCKDfreepend | 4 | Maximum allowable CCKD freespace pending interval value |
| DefCCKDfreepend | -1 | Default CCKD freespace pending interval value |
| MaxCCKDrat | 16 | Maximum allowable CCKD read-ahead tracks |
| DefCCKDrat | 2 | Default CCKD read-ahead tracks |
| MaxCCKDraq | 16 | Maximum allowable CCKD read-ahead queue size |

| Name | Default | Description |
|---|---|---|
| DefCKDraq | 4 | Default CCKD read-ahead queue size |
| MaxCCKDra | 9 | Maximum allowable CCKD read-ahead threads |
| DefCCKDra | 2 | Default CCKD read-ahead threads |
| MaxCCKDwr | 9 | Maximum allowable CCKD writer threads |
| DefCCKDwr | 2 | Default CCKD writer threads |
| MaxCCKDgcint | 60 | Maximum allowable CCKD garbage-collection interval (seconds) |
| DefCCKDgcint | 5 | Default CCKD garbage-collection interval (seconds) |
| MaxCCKDtrace | 200000 | Maximum allowable CCKD trace table size (number of entries) |
| DefCCKDtrace | 0 | Default CCKD trace table size (number of entries) |
| MaxCCKDcache | 64 | Maximum allowable CCKD cache size (MB) |
| MaxCCKDl2cache | 2048 | Maximum allowable CCKD level-2 cache size (MB) |
| MeterThreadRate | 1000 | CPU percent utilization meter update interval (milliseconds) |
| NagleThreadNagleRate | 75 | Maximum 'continue buffering' (to prevent screen refresh) message reception rate (milliseconds) |
| NagleThreadMaxNagleRate | 375 | Maximum allowable delay before displaying buffered messages (milliseconds) |
| MinCaptureBuffsize | 64 | Minimum allowable TunTap32 WinPcap device driver capture buffer size (KB) |
| MaxCaptureBuffsize | 16384 | Maximum allowable TunTap32 WinPcap device driver capture buffer size (KB) |
| DefCaptureBuffsize | 1024 | Default TunTap32 WinPcap device driver capture buffer size (KB) |
| MinPacketBuffsize | 16 | Minimum allowable TunTap32 DLL I/O buffer size (KB) |
| MaxPacketBuffsize | 1024 | Maximum allowable TunTap32 DLL I/O buffer size (KB) |
| DefPacketBuffsize | 64 | Default TunTap32 DLL I/O buffer size (KB) |

**Table 19: Hercules Windows GUI Registry Keys**

# 10. Installation of HercPrt

## 10.1 Downloading the Binaries

The HercPrt utility can be downloaded from http://www.softdevlabs.com/Hercules/hercprt.html. There is a 32-bit and a 64-bit version available. The 64-bit version contains and installs also the 32-bit binaries, so there is no need to download both versions.

## 10.2 Installation Steps

After selecting the correct binaries as described above, double click on the downloaded setup file to start the installation process. A welcome screen appears with displays some general advices for installing the HercPrt utility.



**Figure 54: HercPrt – Welcome Screen**

Click on the "Next" button to continue the installation process.

The next screen displays the HercPrt license agreement.
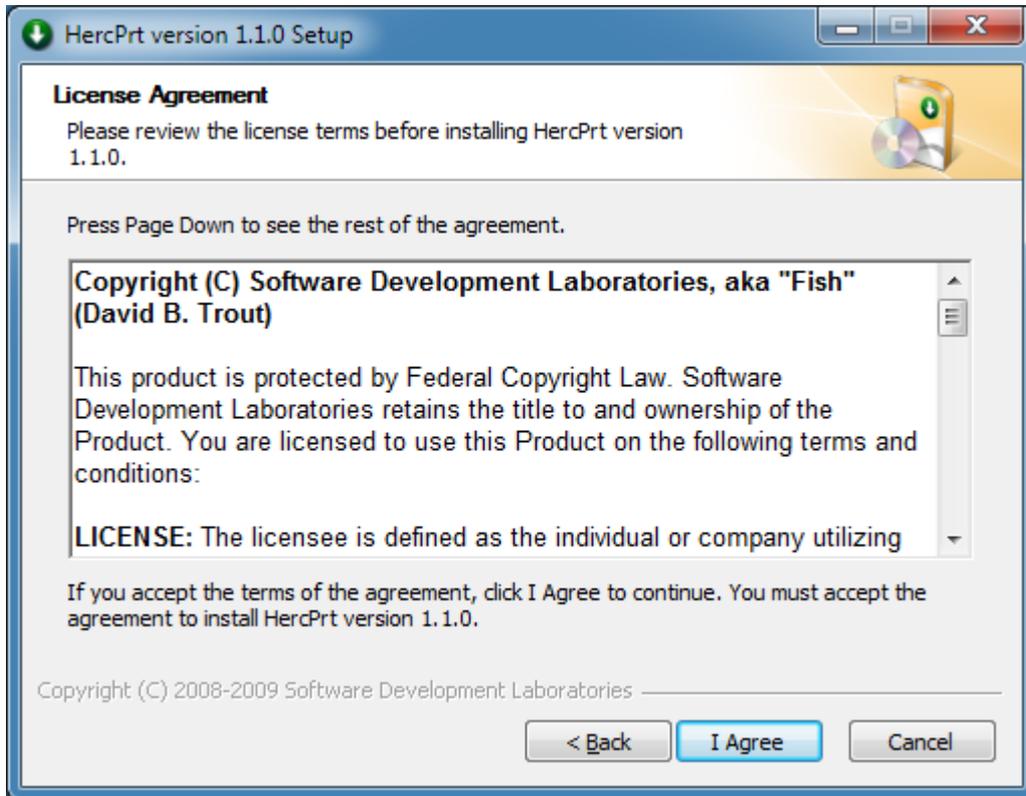


**Figure 55: HercPrt – License Agreement**

Scroll through the license agreement and read it carefully. Next, click on the "I Agree" button to continue with the next step.

In the next screen the "Pre-Installation ReadMe" is shown. This file contains some instructions that is useful to know before actually installing the HercPrt utility. Please read carefully the part about the prerequisites (e.g. "FishLib" etc.).
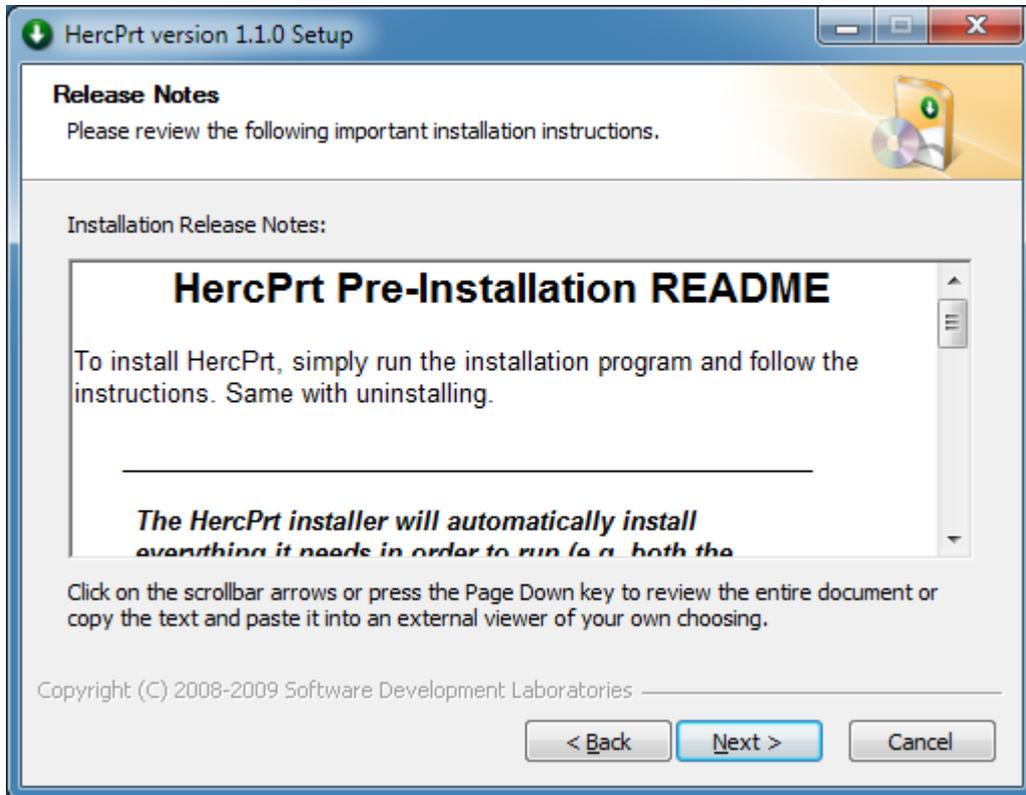


**Figure 56: HercPrt – Pre-Installation ReadMe**

If you have read the instructions, then click on the "Next" button to proceed.

The installation dialog now presents the section where normally the components to install are chosen. Because HercPrt consists of only the main binaries you are not able to actually choose something here. The check box to deselect the main binaries is greyed out.
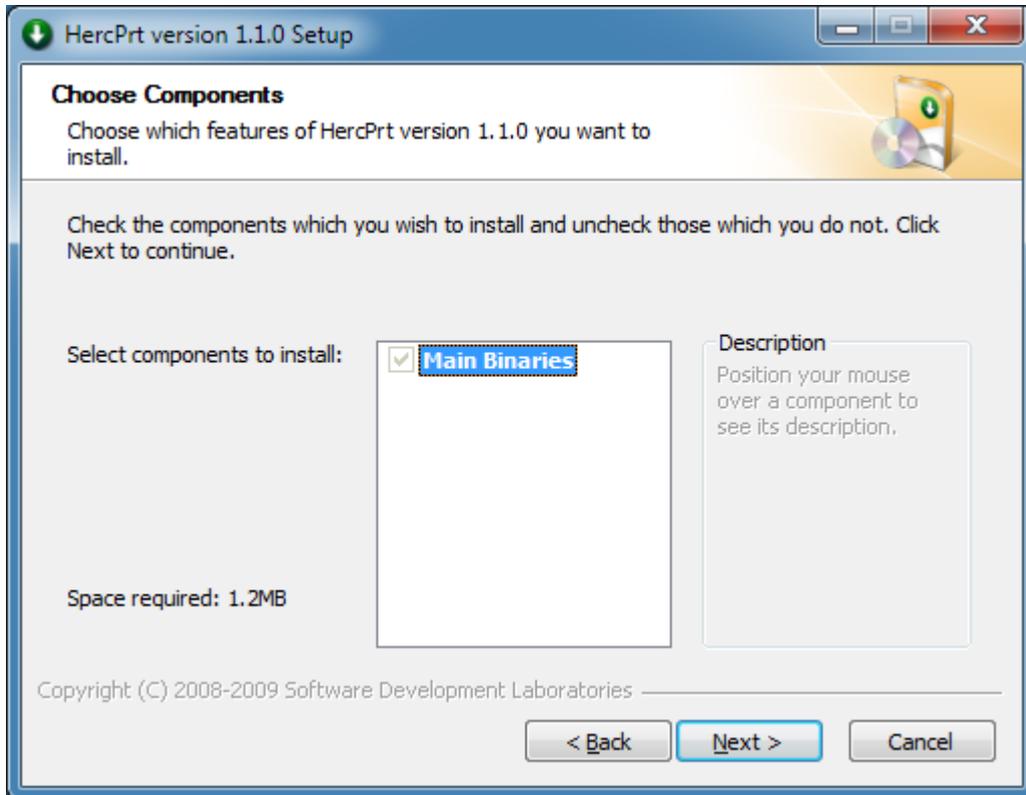


**Figure 57: HercPrt – Choose Components**

Click again on the "Next" button to continue with the installation dialog.

On the next screen you can specify the destination folder for the program files. You can work with the re-commended installation path or you can choose a destination on your own. Clicking on "Browse" you are presented with a standard Windows dialog, for selecting another destination folder.
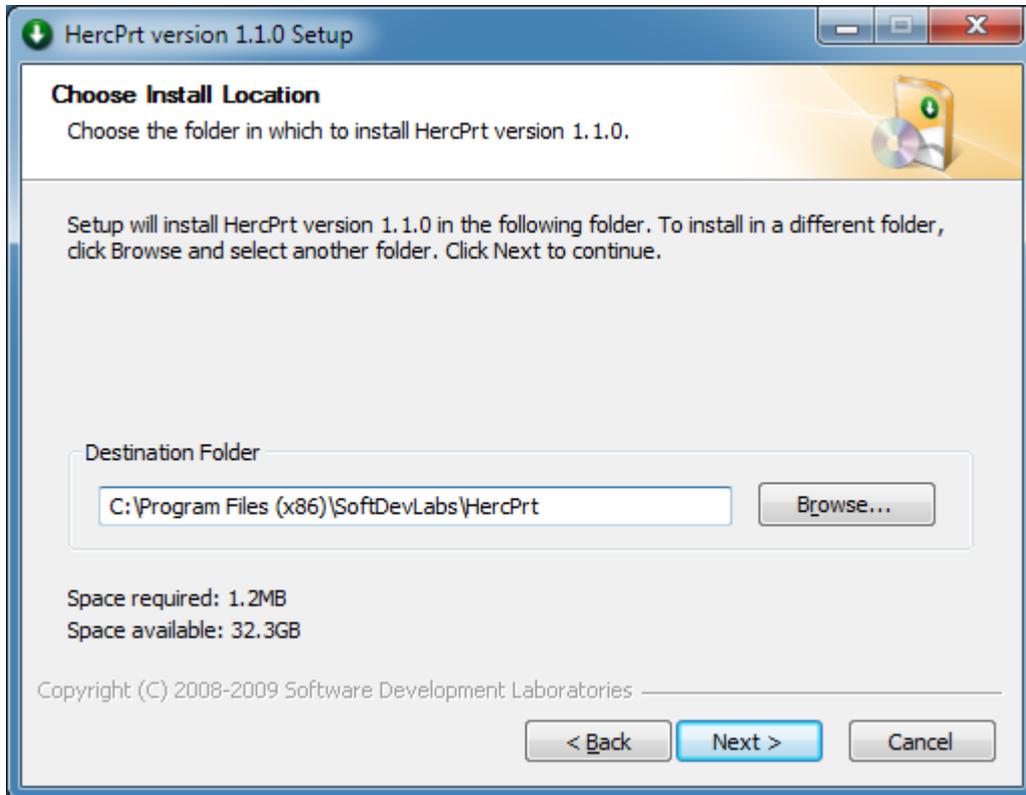


**Figure 58: HercPrt – Choose Install Location**

When you are satisfied with your choice of the destination folder then click on the "Next" button to continue with the next step.

On this screen you can select the start menu folder in which you would like to create the program's shortcuts. As per default the shortcuts will be created under "SoftDevLabs\HercPrt".
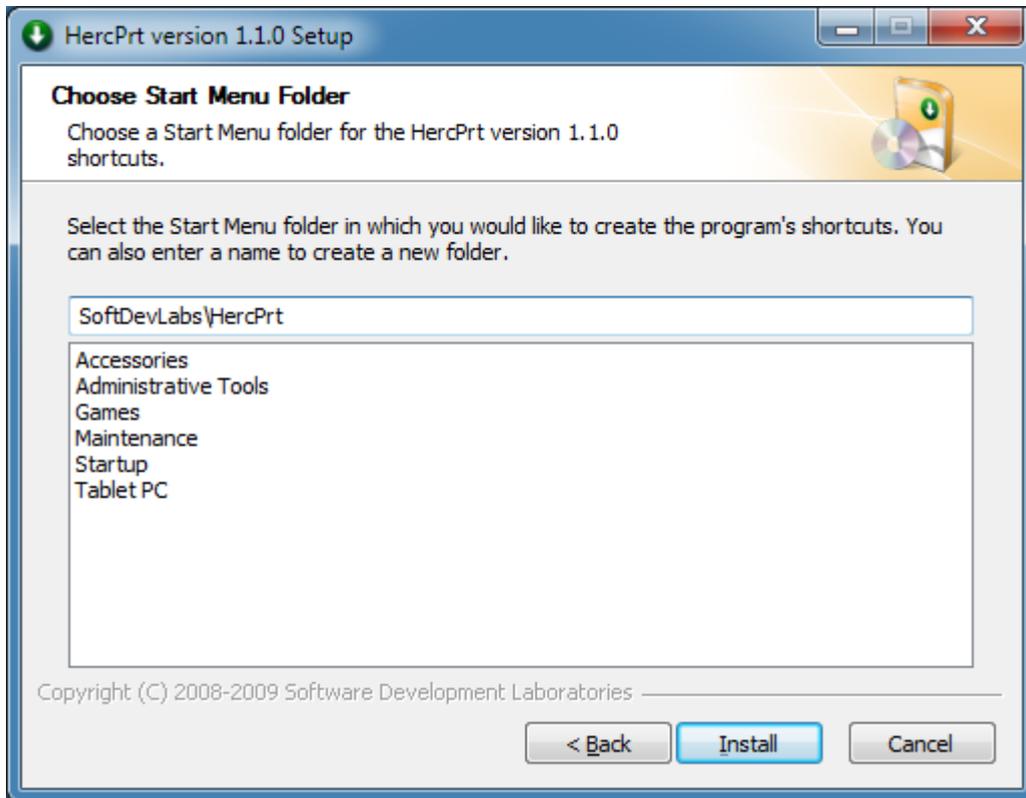


**Figure 59: HercPrt – Choose Start Menu Folder**

Clicking on the "Install" button the actual installation begins.

A scrolling window shows the detailed log about the running installation and a green bar shows the progress of the installation process. After the installation is completed this windows remains on the screen, so you can scroll through and read the installation log.
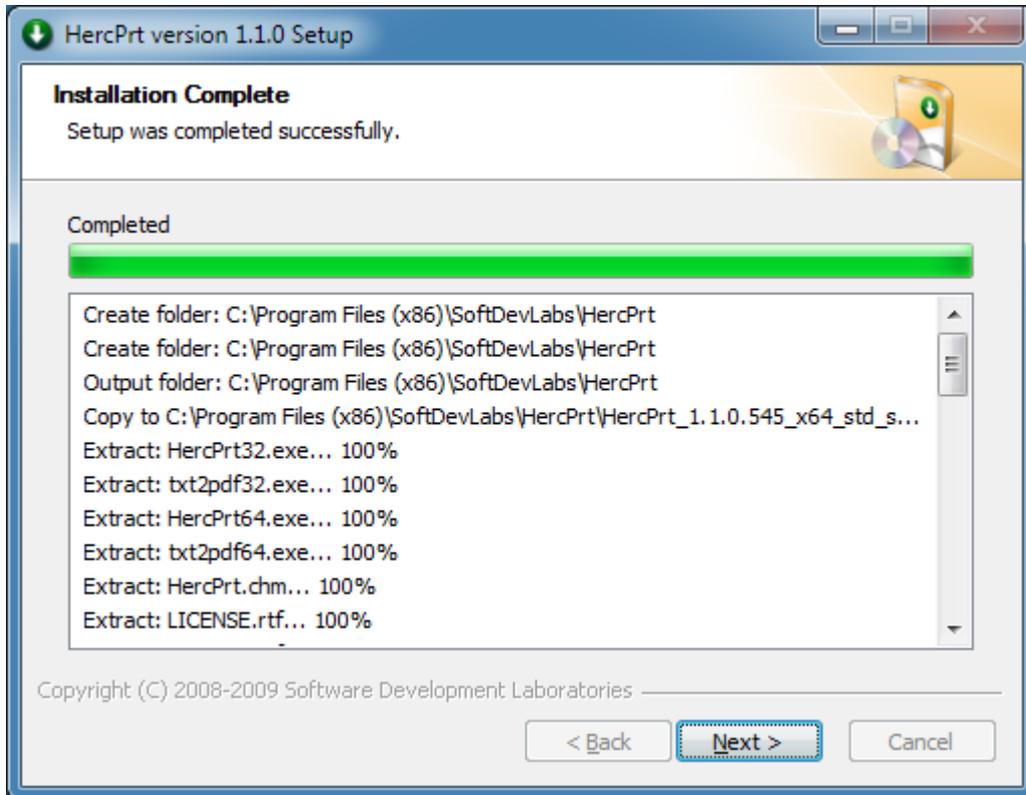


**Figure 60: HercPrt – Installation Progress Bar**

Click on the "Next" button to proceed with the last step of the installation.

This is the final window of the installation. You have the possibility to have shown a "ReadMe" file containing the latest information about HercPrt when exiting the installation dialog. If you are not interested in having this information presented then deselect the "Show Readme" check box.
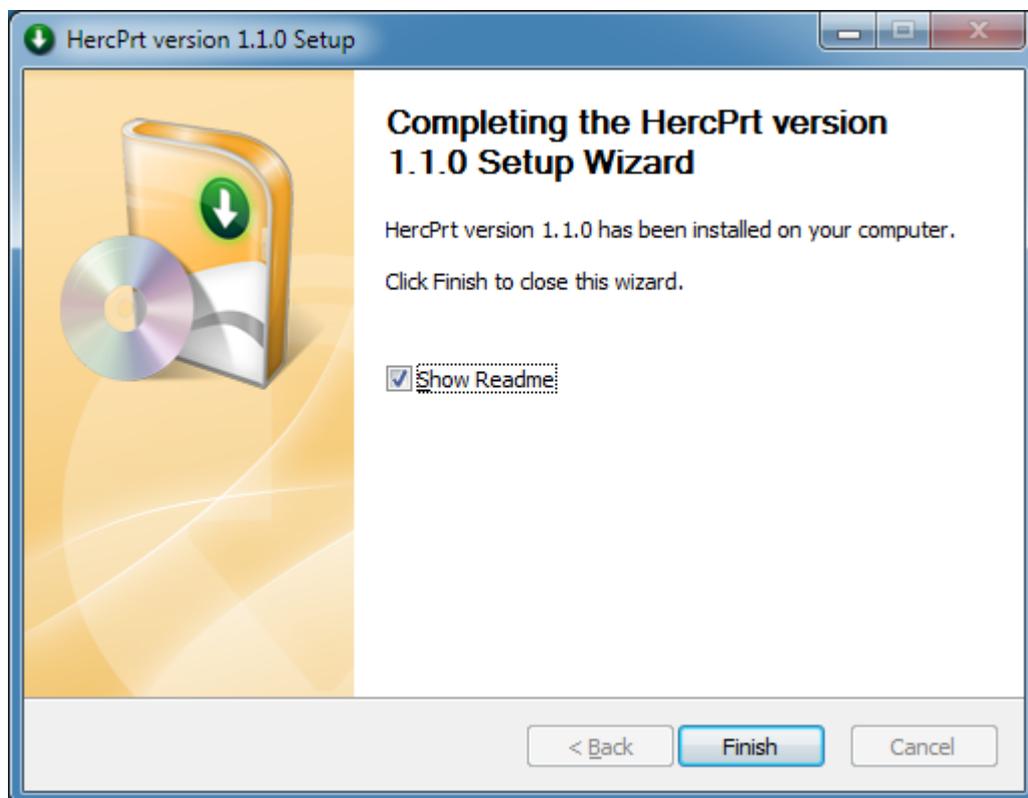


**Figure 61: HercPrt – Completing the Setup Wizard**

After clicking on the "Finish" button the installation dialog is terminated. The HercPrt utility is now installed successfully on your computer. To start the utility look in the Windows start menu under the menu point you just selected some steps before. If you installed the 32-bit version of HercPrt you will find only an entry to "HercPrt32", if you however installed the 64-bit version you will see an entry to "HercPrt64" as well as an entry to "HercPrt32".

## 10.3  Starting HercPrt

The HercPrt utility is started through clicking on the "HercPrt32" or "HercPrt64" entry in the Windows start menu folder you have chosen through the installation process.

## 10.4  Customization Steps

This section covers only the most important customization steps. A more detailed information that covers all aspects of the HercPrt utility can be found in the Hercules "Utilities Guide".

## 10.4.1 Hercules Definitions

On the Hercules side some changes in the configuration file are needed to be able to use HercPrt. In the device definition section you will find probably some entries for currently defined printers, similar to these statements:

```
devaddr devtype filename options
```

Example:

```
000E 1403 D:/MVS/Printer/PRT00E.TXT CRLF NOCLEAR
000F 1403 D:/MVS/Printer/PRT00F.TXT CRLF NOCLEAR
```

These statements have to be changed to:

```
devaddr devtype ipaddress:port sockdev
```

Example:

```
000E 1403 192.168.0.101:14031 sockdev
000F 1403 192.168.0.101:14032 sockdev
```

The value '*ipaddress:port*' is the TCP/IP address and the port number at which the Hercules socket printer will listen for incoming connections. The IP address is typically the IP address of the Windows system where HercPrt is running on and the port number can be any value from 1024 to 65535.

Note that Hercules socket printers do not support any other options besides "SOCKDEV", the "clrf" and "noclear" options for example are invalid when defining a socket printer. For details on how to define printers in a Hercules configuration file please see the Hercules "User Reference" manual.

## 10.4.2 HercPrt Program Options

In order to use HercPrt at least the following definitions have to be made in the "Program Options Panel" (see figure below). All other fields can be left unchanged before the first usage of the utility.

Detailed information about configuring HercPrt, including both the "Program Options Panel" and the "PDF Options Panel" can be found in the Hercules "Utilities Guide".

**Printer ID**
This field is used to enter a descriptive name for the Hercules printer you are defining. The value can be anything that uniquely identifies the printer being defined. If you already have some printers defined you can select one of these previously defined printers from the dropdown list to automatically populate the remaining controls with the values for the chosen printer.

**IP Address**
This must be the IP address where your Hercules printer is listening for incoming connections.

**Port Number**
This must be the port number where your Hercules printer is listening for incoming connections.

**Control File**
This is the filename of the "Job Separator Control File" for the printer. The job separator control file defines what your Hercules guest operating system's job separator pages look like and allows HercPrt to detect where one print job ends and the next

print job begins.

This allows HercPrt to break spooled output into separate Windows files, one for each printout, and to name the files using information from the job accounting fields extracted from the print fields on the actual job separator page itself.

Click on the arrow to display a standard Windows file open dialog where you can select the job separator control file you want to use for this printer.

**Spooler Dir**        Enter here the name of the directory where you want the print files to be placed.

Click on the arrow to display a standard Windows file open dialog where you can select the directory for the print files of this printer.
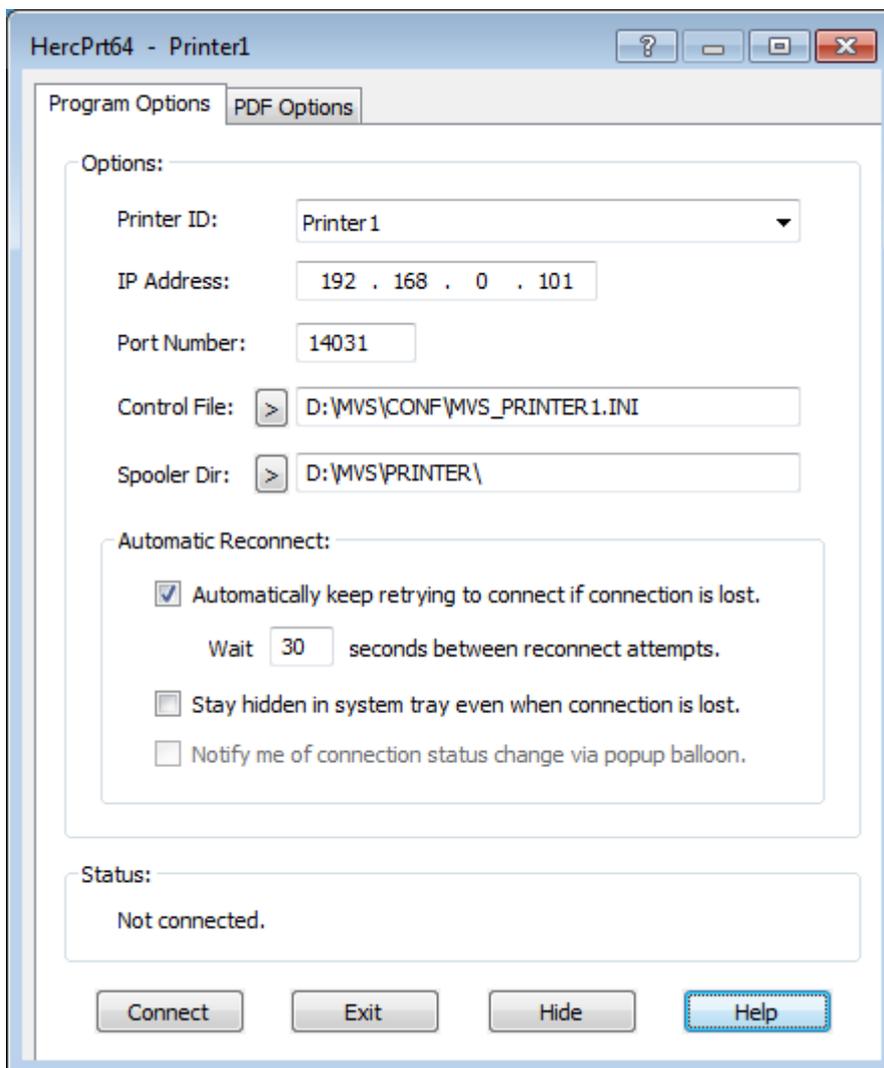


**Figure 62: HercPrt Program Options Panel**

# 11. Installation of CTCI-W32

## 11.1 Downloading the Binaries

The CTCI-W32 components can be downloaded from http://www.softdevlabs.com. Two packages are required, CTCI-W32 and FishLib.

CTCI-W32 consists of FishPack.dll, TunTap32.dll and TT32Test.exe packaged together as one product. The FishLib package is required with all recent versions of Fish's software. It contains common routines used throughout the packages.

Note: Beginning with release 3.2.1.160 of CTCI-W32 additional DLLs are required. These are Microsoft MFC and VC Runtime DLLs that you can download from the address mentioned above. The installation takes a few seconds and does not require a reboot. There are a 32-bit and a 64-bit version of these DLLs. Please ensure you are using the correct one according to the product you are installing (32-bit or 64-bit version of the Windows GUI).

If you previously installed the Hercules Windows GUI as described earlier in this manual then these DLLs are already present on your system. Note that you only need to install these C Runtime DLLs once even if new versions of CTCI-W32 are subsequently installed.

## 11.2 Installation Steps

The installation of the CTCI-W32 packages is straightforward. Unzip the executables and DLLs from the downloaded ZIP files into the same directory as the Hercules executables and the installation is complete.

## 11.3 Customization Steps

The customization of the CTCI-W32 consists of three major steps described further below:

- Configuring Windows Networking
- Configuring Hercules
- Configuring the Guest Operating Systems TCP/IP Settings

For details on how CTCI-W32 works internally see the Hercules "General Information" manual.

### 11.3.1 Configuring Windows Networking

Only minor configuration of Windows networking is required in order to use CTCI-W32, beginning with WinPcap (see chapter 7). Next verify that the network adapter has an IP address assigned and default gateway assigned. In most installations this will already be the case and no further configuration will be required.

If you are using DHCP rather than assigning a static IP address to the network card, then it will be necessary to tell Hercules the exact hardware (MAC) address of the network adapter Hercules is to use. See section "Configuring Hercules" below for further information.

To verify the IP address of your network interface open the network card properties and double-click on the "Internet Protocol (TCP/IP)" component. The following properties dialog will appear:
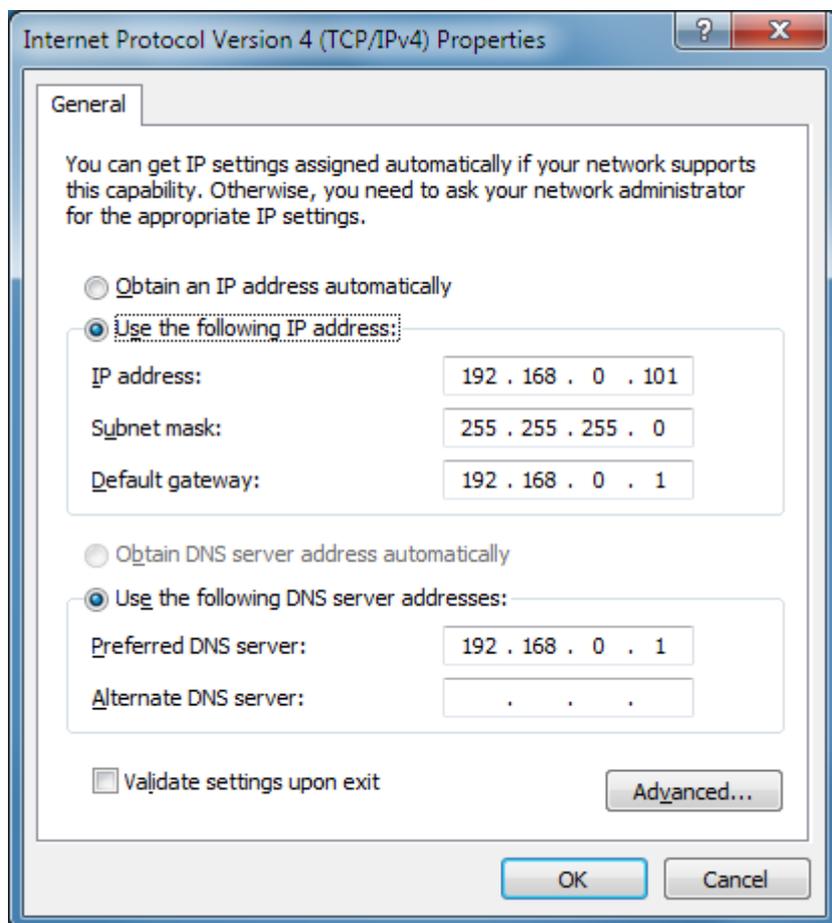


**Figure 63: Windows TCP/IP Properties**

Make sure you have entered a valid IP address, subnet mask and default gateway. This is all that is necessary on the Windows system side, the remaining configuration of the CTCI-W32 functionality is controlled via Hercules device statements.

## 11.3.1.1 IP Forwarding

You may or may not need to have "IP Forwarding" enabled on your Windows system. Whether this is required or not depends on your use of a router within your network. If you are using a router to define routes to hosts on your LAN, then you should not need "IP Forwarding" enabled on your Windows system. If however you are not using a physical hardware router then you will likely need to enable IP Forwarding.

To allow your Hercules guest operating system to communicate with hosts on the LAN apart from the host machine itself (where Hercules is running), TCP/IP requires routes to and from the Hercules hosted operating system. This is necessary so that the Hercules virtual guest OS's packets can be properly routed to their final destination. This is typically the role performed by a hardware router. If you do not use a hardware router then Windows "IP Forwarding", together with appropriate ROUTE statements, will perform this role.

Without a router or "IP Forwarding" enabled you will only be able to communicate with your Hercules guest OS from the same Windows computer that Hercules is running on and the hosted OS will only be able to communicate with the Windows system it is running on.

To enable "IP Forwarding" on Windows, make the following registry change:

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters]

"IPEnableRouter"="1"
```

**Figure 64: Windows "IP Forwarding" Registry Key**

If the "IPEnableRouter" (or "EnableRouting") registry value does not yet exist under the mentioned registry key then you will have to add it manually. Select 'New' from the 'Edit' menu and add a new DWORD value with the indicated name.

## 11.3.2  Configuring Hercules

Two IP addresses must ultimately be assigned, one for the Hercules end of the link and another for the driving system's (host systems) end of the link.

Note that the format of the Hercules configuration file statements for networking emulation has changed from previous releases of Hercules. The older 'CTCI-W32' format has been deprecated. Please ensure your CTC device statements are updated to use the newer 'CTCI' or 'LCS' format in order to remain compatible with current and future releases of Hercules. For example:

```
0E20  3088  CTCI  192.168.0.4  192.168.0.2
0E21  3088  CTCI  192.168.0.4  192.168.0.2
```

**Figure 65: Sample CTCI definition for static IP addresses**

The first IP address (192.168.0.4) is the IP address of your Hercules system, i.e. the guest operating system running under Hercules. The second IP address (192.168.0.2) identifies the CTCI-W32 logic, i.e. the network adapter that CTCI-W32 is to use in order to reach the Windows TCP/IP stack.

It is recommended that the IP address you choose for your virtual guest (like 192.168.0.4 in the above example) is in the same subnet as your Windows host in order for CTCI-W32 to work properly. The simplest way to approach things is to configure your guest as if it were a real system connected to your network. Thus it would normally be assigned an IP address within the same subnet as all the other workstations on your LAN.

Note that it is possible to place the guest OS in a separate subnet to your Windows host if you are prepared to define the proper routing entries. However it is usually a lot simpler and less problematic to define all hosts in the one IP subnet.

If your network adapter does not have a static IP address then instead of specifying an IP address as the second parameter in the Hercules device statement, you must specify the MAC address of the adapter you wish to use:

```
0E20  3088  CTCI  -n 00-80-B3-E1-DF-69  192.168.0.4  0.0.0.0
0E21  3088  CTCI  -n 00-80-B3-E1-DF-69  192.168.0.4  0.0.0.0
```

**Figure 66: Sample CTCI definition for dynamic IP addresses**

Note that the format of the CTCI definition statement is slightly different in this case. The '-n' parameter informs Hercules that this is a MAC address, however the parser still expects to see two IP addresses, hence the second address is specified as a 'dummy' address, i.e. all zeroes. The first IP address is still the address you have assigned to the OS hosted by Hercules.


## 11.3.3 Configuring the Guest Operating System TCP/IP Settings

The procedure for configuring TCP/IP on any guest operating system you plan to run under Hercules is different for each OS. For example the procedure is completely different for z/VM than it is for z/OS. For detailed instructions on how to configure TCPIP profiles for using a CTCI device please refer to the appropriate operating system manuals.

A sample configuration is shown in below:

```
TCP/IP configuration using CTCI-W32
-----------------------------------

The following values belong in the TCP profile.

The first entry in the TCP profile is the DEVICE entry.  This is the Para-
meter that defines the UCB address assigned to the interface within the MVS
environment.  It is related to the LINK statement to build the TCPIP connec-
tion between the hardware and the TCPIP stack.

        DEVICE     CTC1        CTC          E20
        LINK       CTC1L       CTC          0          CTC1

The second entry is the HOME entry.  This is where the IP address is
assigned to the device interface within the TCPIP stack.  The HOME entry
connects the IP address to the hardware through the LINK defined with the
DEVICE.

        HOME
        192.168.0.4    CTC1L

The GATEWAY describes how the IP stack gets to the rest of the network.  It

defines the physical first hop from the local interface out to the entire
network.  In our example, we are defining a gateway to one physical IP


address, 192.168.0.2.  1492 is the data packet size.

        GATEWAY
        192.168.0.2    =       CTC1L    1492     HOST

The DEFAULTNET statement instructs what path the IP stack should take to
```

```
look for address it doesn't know about.  IP assumes the next hop will be
able to resolve the IP address the stack is trying to connect to.

        DEFAULTNET 192.168.0.2 = CTC1L 1492 0

The final statement needed in the profile is the command to instruct TCPIP
to start the device.

        START CTC1
```

**Figure 67: Sample TCP/IP Configuration for CTCI-W32**

If you wish to try using an LCS (LAN Channel Station) device instead then the following sample may be useful. LCS can handle any Ethernet packet rather than just IP packets:

```
Sample regular / normal LCS device definitions

0E20-0E21  LCS  -n  172.16.9.163   -m  00-00-5e-90-09-5d   172.16.9.93

-------------------------------------------------------------------

Sample LCS device definitions for Enterprise Extender

0E20-0E23  LCS  -n  172.16.9.163   -o  oatfile.txt

-------------------------------------------------------------------

oatfile.txt:

*********************************************************
* Dev  Mode Port Entry specific information            *
*********************************************************
0E20  IP   00   PRI   172.16.9.93
HWADD      00         00-00-5E-90-09-5D

0E22  IP   01   SEC   172.16.10.93
HWADD      01         00-00-5E-90-0A-5D

-------------------------------------------------------------------

Sample TCPIP PROFILE statements for Enterprise Extender




(Note: not all statements are shown)

IPCONFIG  DATAGRAMFWD  VARSUBNETTING  SYSPLEXROUTING

DEVICE  LCS1     LCS      E20   AUTORESTART
LINK    ETH1     ETHERNET 0     LCS1
```

```
DEVICE  VDEV1      VIRTUAL    0
LINK    VLINKA     VIRTUAL    0     VDEV1

DEVICE  IUTSAMEH MPCPTP
LINK    EELINK    MPCPTP           IUTSAMEH

START   LCS1
START   IUTSAMEH

HOME
172.16.9.93    ETH1
172.16.10.93   VLINKA

BEGINROUTES
ROUTE   172.16.0.0   255.255.0.0   =              ETH1   MTU 1492
ROUTE   DEFAULT                    172.16.13.1    ETH1   MTU 1492
ENDROUTES

PORT
12000 UDP VTAM
12001 UDP VTAM
12002 UDP VTAM
12003 UDP VTAM
12004 UDP VTAM

-------------------------------------------------------------------

(Not shown: VTAM definitions)
```

**Figure 68: Sample LCS Configuration for CTCI-W32**

## 11.3.3.1  Defining the Guest's Default Gateway

The correct definition of your guest operating systems default gateway depends on whether or not you are using a real router. If you do not have a real router and are instead using the Windows "IP Forwarding" (IP Routing) feature to perform routing, then your default gateway should be the physical adapter on your Windows system that your virtual interface is using.

Alternatively if you do have a real router then your default gateway should be the IP address of your actual router. In the sample LCS configuration immediately above 172.16.13.1 is a real network router and thus knows how to route traffic to the other end of the guest's Enterprise Extender link. Note that, as recommended, the guest's IP addresses (172.16.9.93 and 172.16.10.93) are within the same subnet (255.255.0.0) as the Windows host that Hercules is running under (172.16.9.163).

## 11.3.4 Tweaking CTCI-W32

The CTCI-W32 protocol supports some additional tuning parameters in addition to the required parameters. You can adjust the size of the WinPcap kernel device driver's internal packet buffer, as well as the size of TunTap32.dll's own packet I/O buffer to try and increase the performance of your network:

```
0E20  3088  CTCI  -n 00-80-B3-E1-DF-69  -k 1024  -i 64  192.168.1.99  0.0.0.0
0E21  3088  CTCI  -n 00-80-B3-E1-DF-69  -k 1024  -i 64  192.168.1.99  0.0.0.0
```

**Figure 69: CTCI-W32 Tuning Parameters**

The numbers 1024 and 64 in the above statements are the size of the WinPcap kernel device driver's internal packet buffer (in KB), and the size of TunTap32 DLL's internal packet I/O buffer (in KB) respectively. Please note that memory for WinPcap's kernel device driver's packet buffer is taken from Windows "non-paged" memory pool (i.e. from your system's real physical memory). The size of this buffer has a direct impact on Windows performance. If you choose a ridiculously large number here Windows will have little memory left to work with and the overall performance of your Windows system will be degraded.

The second number (64 in the above example) is the size in KB of the I/O buffer allocated inside the TunTap32 DLL. This defines how much data TunTap32 will request from the device driver each time it needs to do an I/O operation to the physical adapter and therefore how much data will be transferred from the device driver's internal buffer to TunTap32's I/O buffer. The memory for this buffer is allocated in virtual memory from the Hercules process's address space and should not impact Windows's performance unless a ridiculously large number is specified. If too high a number is specified then Windows will likely 'thrash' attempting to page the entire Hercules address space and performance will be degraded.

The TunTap32 DLL passes packets to Hercules one at a time.  When it runs out of packets in its internal I/O buffer it requests more data from the WinPcap device driver via the FishPack DLL. The TunTap32 I/O buffer size determines how much data and therefore how many packets it will receive from WinPcap for each request.

The larger this buffer, the fewer actual I/O's TunTap32 will perform to WinPcap via FishPack, but at the same time the longer the interval between those I/O's.  This implies that the WinPcap device driver will have to buffer its data for a longer period between less frequent I/O's. This requires a larger kernel buffer in order to ensure that no packets are lost during periods of high network activity. Conversely, the larger the actual I/O (i.e. the more bytes transferred per I/O), the longer each I/O takes. Although this is measured in microseconds, the delay holds up the entire Windows operating system during transfers of data from kernel-space to a user-space. It is good to perform as little I/O as possible as these can be expensive, but the benefit drops quickly if each I/O has significant impact on the overall system, so choose these settings carefully.

Unless you are using a Gigabit or faster network it is usually best to leave these at their default settings. The currently implemented minimum, maximum and default values are:

| Buffer Type | Minimum | Default | Maximum |
|---|---|---|---|
| WinPcap device driver capture buffer size | 64 KB | 1 MB | 16 MB |
| TunTap32 DLL I/O buffer size | 16 KB | 64 KB | 1 MB |

**Table 20: CTCI-W32 Buffer Sizes**

If you decide to adjust these buffer size values to try and increase network throughput and performance, you may find the Hercules "tt32stats" command useful.

The tt32stats command shows the actual tt32 statistics as shown below:

```
23:59:11.098 0000066C tt32 stats e20
23:59:11.098 0000066C TunTap32.dll Statistics:
23:59:11.118 0000066C Size of Kernel Hold Buffer:      1024K
23:59:11.118 0000066C Size of DLL I/O Buffer:            64K
23:59:11.128 0000066C Maximum DLL I/O Bytes Received:     2K
23:59:11.138 0000066C            7 Write Calls
23:59:11.148 0000066C            8 Write I/Os
23:59:11.148 0000066C          319 Read Calls
23:59:11.158 0000066C          259 Read I/Os
23:59:11.168 0000066C          282 Packets Read
23:59:11.168 0000066C            8 Packets Written
23:59:11.178 0000066C        38172 Bytes Read
23:59:11.178 0000066C          542 Bytes Written
23:59:11.188 0000066C            1 Internal Packets
23:59:11.198 0000066C            2 Ignored Packets
```

**Figure 70: tt32 Statistics**

If the reported "Maximum DLL I/O Bytes Received" value is identical to the value specified for the "Size of DLL I/O Buffer", then each time TunTap32 requested more packets the WinPcap device driver had at least a full buffer worth waiting to be delivered to TunTap32.

This generally indicates that the default buffer size is too small or that your network is extremely busy. This would be the least common situation though. TunTap32 is rarely unable to deliver an entire buffer of packets to Hercules before another buffer arrives unless Hercules is performing extremely poorly or your network really is under heavy load.

# 12. Installation of Vista tn3270



**Figure 71: Vista tn3270 Logo**

## 12.1 Vista tn3270

Vista tn3270 is a Windows program designed to emulate IBM 3270 terminals connected to a host via an IP. It is written by Tom Brennan, is currently available as a free 30 day trial and a perpetual license costs approximately $30 US dollars. This emulator was created with mainframe programmers in mind and has some unique features unavailable on even the most expensive commercial emulators.

Vista has features designed especially for programmers such as built-in multiple cut and paste buffers, fully customizable keyboard, extensive select/copy/paste functions – especially the "SelectJCL" function that is used to select dataset names, parameters, and similar items with a single mouse click.

Vista uses bitmapped raster fonts for the clearest text possible. There are 2 sets, "Thick" and "Thin", in 73 sizes each from 4x6 to 16x36. With so many sizes you can easily setup Vista to suit your monitor size and terminal model preferences in either full-screen or windowed mode.

*Parts of the following sections about the Vista tn3270 installation have been taken from the original Vista tn3270 documentation with the kind permission of Tom Brennan.*

## 12.2 Downloading the Installation Routine

The binaries for the Vista tn3270 Terminal Emulation can be downloaded directly from Tom Brennan's Vista tn3270 webpage using the following link:

www.tombrennansoftware.com

The downloaded version of Vista tn3270 can be used for a free trial for 30 days following installation. After this free trial period a license key must be entered for continued use of the software.

## 12.3 Install Vista tn3270

To start the installation process, just double-click on the downloaded executable file.

The installation welcome screen is presented:
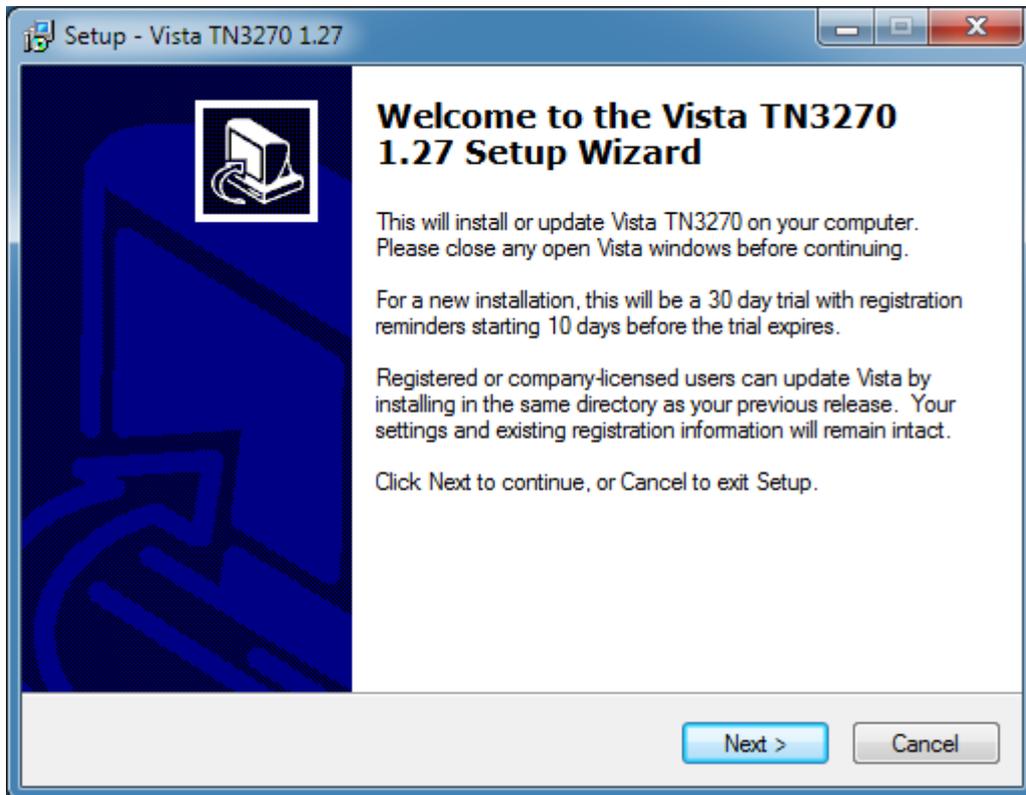


**Figure 72: Vista tn3270 – Welcome Screen**

Click on "Next >" to continue the installation process.

The following screen prompts you for the installation directory. Select a destination directory of your choice and click "Next" to continue.
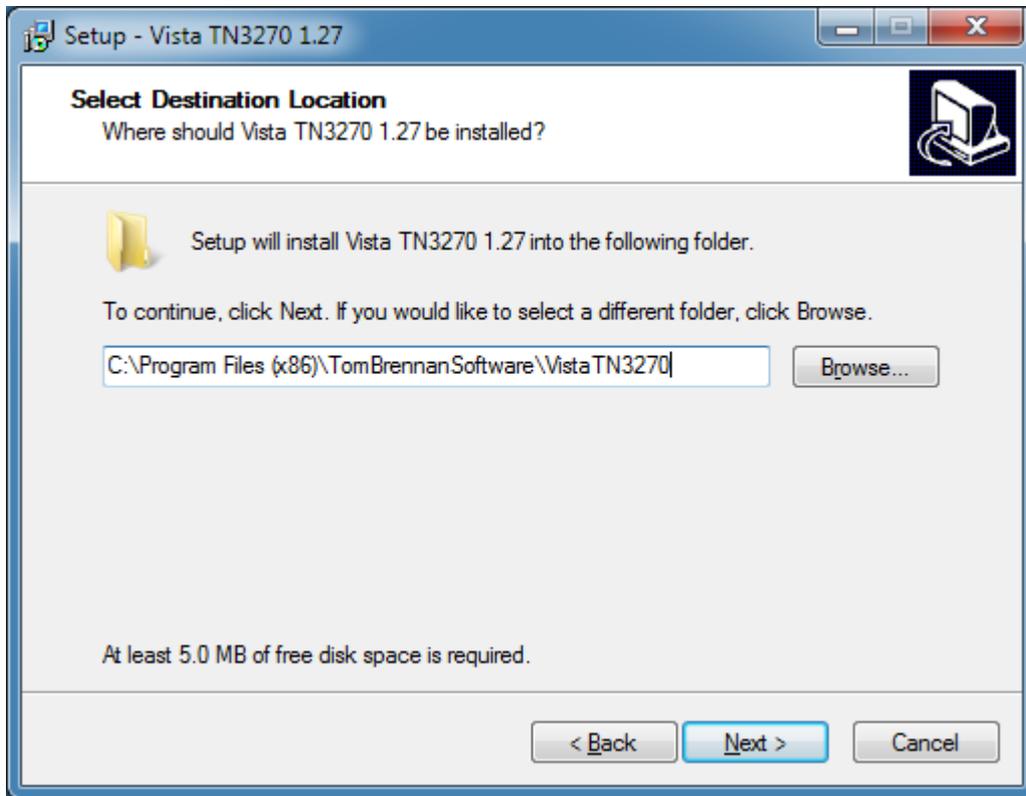


**Figure 73: Vista tn3270 – Select Destination Directory**

The screen presented next allows you to choose the Windows Start Menu Group to which the emulator icons should be added. The default group is usually acceptable. Click on "Next" again to continue.
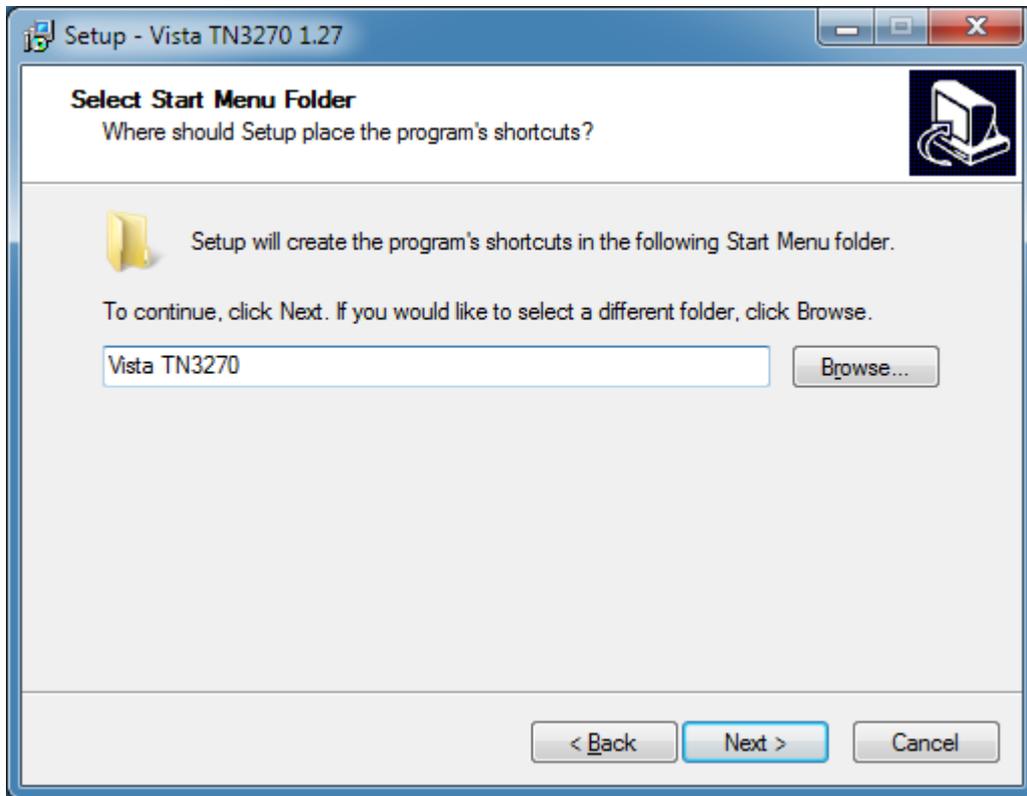


**Figure 74: Vista tn3270 – Select Program Group**

Now the installation program is ready to copy the necessary files to your harddisk. A confirmation screen appears where you can change your previous selections. If you do not want to change any of these click on "Install" and the setup program begins the actual installation.
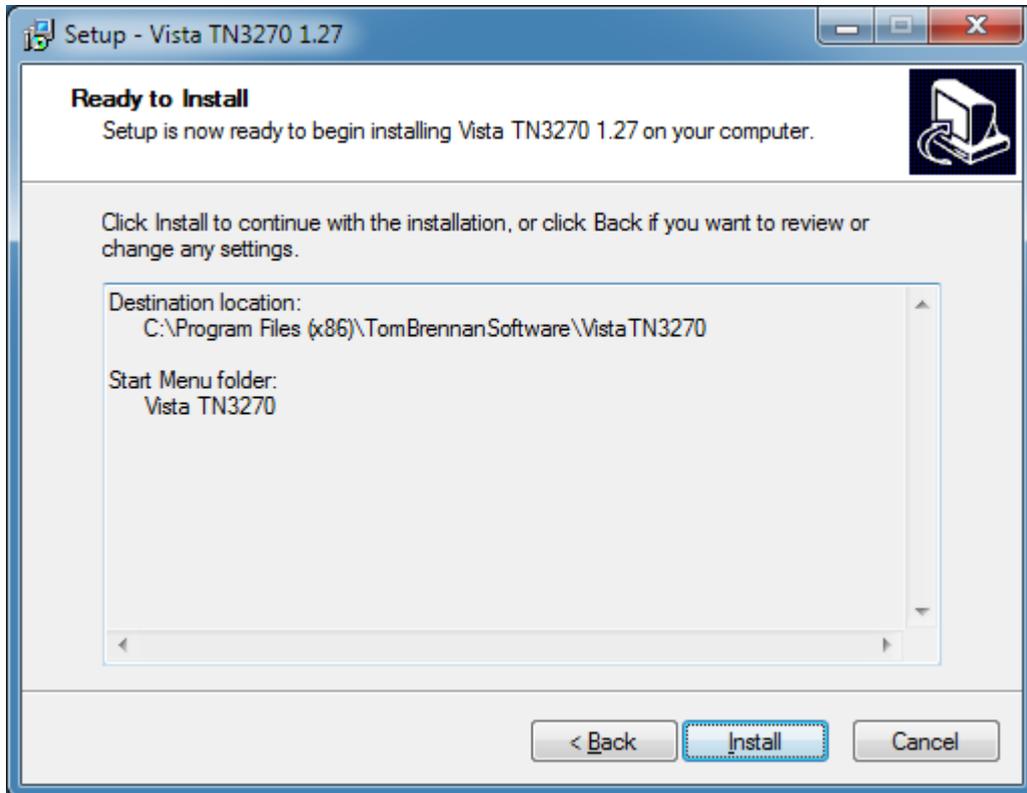


**Figure 75: Vista tn3270 – Ready to Install Screen**

Next the setup program installs Vista tn3270 according to your previous settings.
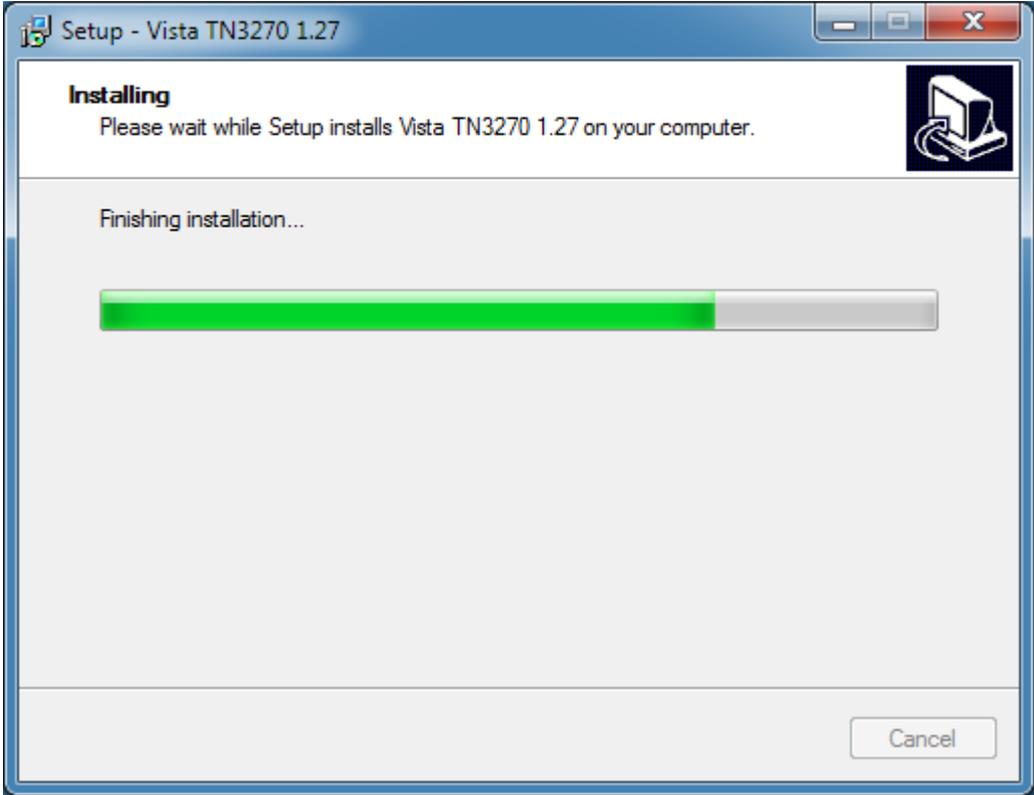


**Figure 76: Vista tn3270 - Installation Progress**

When the installation process is finished the "Setup Completed" screen will appear. You must confirm by clicking "Finish".
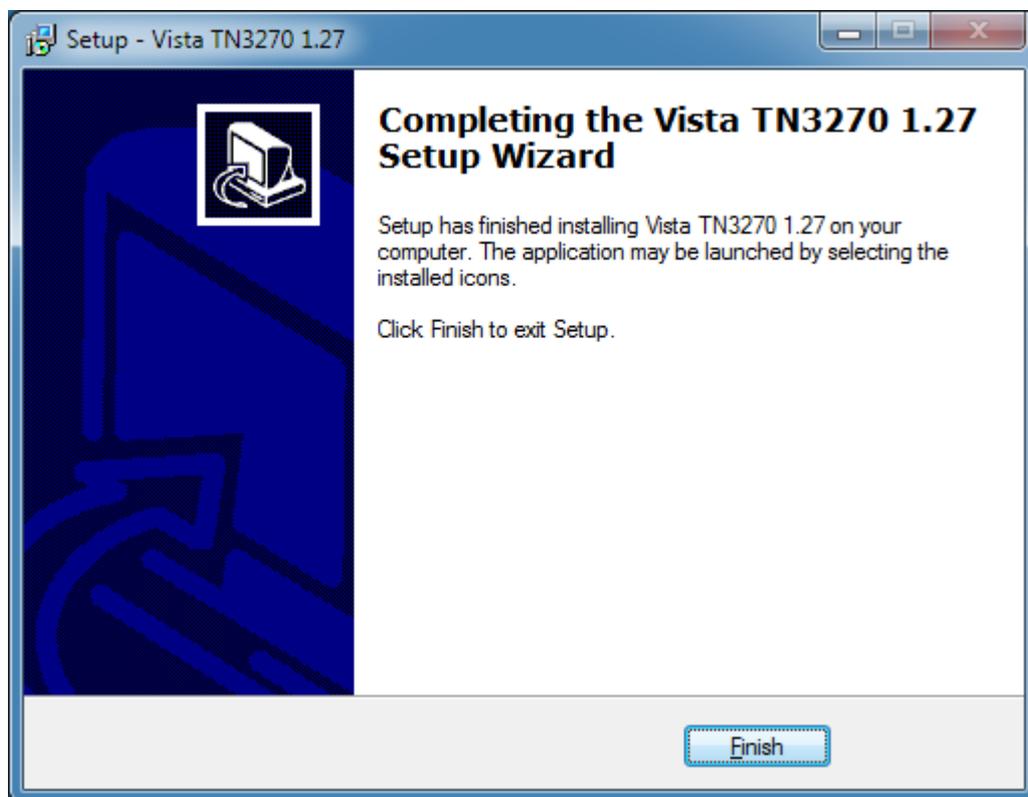


**Figure 77: Vista tn3270 – Setup Completed**

## 12.4 Activation of the Software

The Vista tn3270 software creator, Tom Brennan, allows you to try the product for 30 days. When this period expires a licence key must to be entered to enable the product for use again. You can purchase the product via the online purchasing system at the product website. After purchase a license key is sent by email, normally within 24 hours.

## 12.5 Create Sessions

The detailed process of creating terminal emulation sessions and the impact of all possible options within Vista tn3270 is beyond the scope of this manual. For details regarding these please refer to the printed Vista tn3270 documentation or the online help within the product. A short introduction to creating Vista sessions for connection to a Hercules hosted OS follows.

Start a Vista tn3270 window by clicking on "Vista" in the Vista program folder or wherever you chose to install the product. This creates a new Vista instance and immediately opens the "Start a new Terminal Session" dialog, where you can specify IP address and port number. If instead you click on "Vista Standard Session", then Vista immediately attempts to connect to the last used IP address and port.
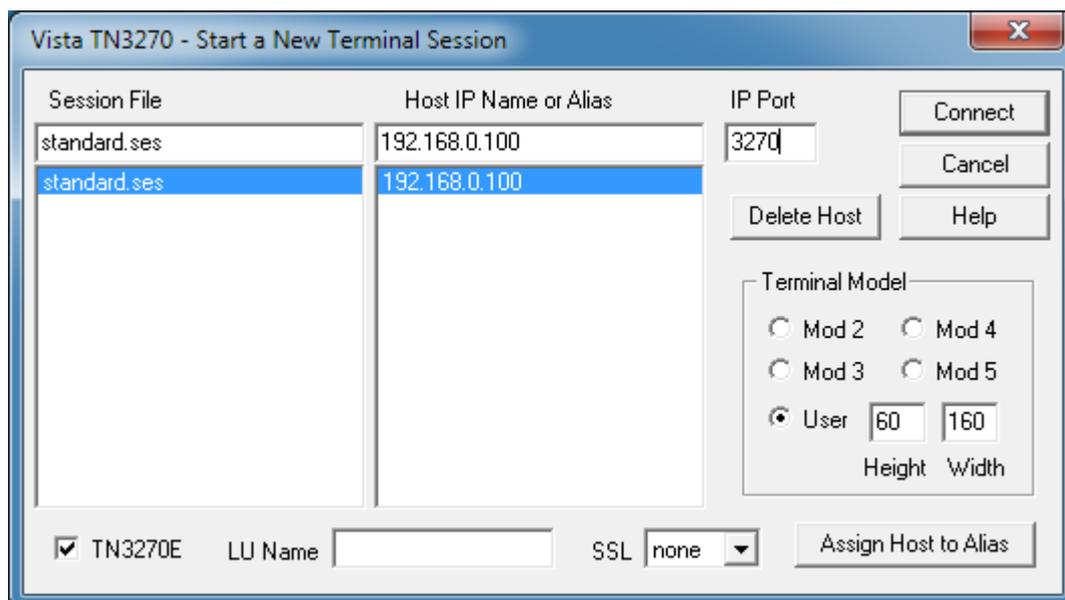
**Figure 78: Vista tn3270 - New Terminal Session Dialog**

The following options can be specified in this dialog:

**Session File**            Select an existing session file or type a new name. A session file contains most
                            of the parameters for a session such as font size, screen colours and other
                            options. Multiple Vista windows can share the same session file but that each
                            window stores parameters when it is closed. This means that the parameters will
                            be set to those of the last window closed.

**Host IP Name/Alias**      This name is usually a dot address like 206.85.100.23 or a DNS (Domain Name
                            Server) name, such as "tn3270.company.com". It can also be an Alias name that
                            points to either a DNS or dot address (see Assign Host to Alias below). IP names
                            and their associated port numbers are stored in the VISTA.INI file rather than the
                            session file so that they can be shared among sessions.

                            The special host name "localhost" or the IP address "127.0.0.1" means the local
                            machine.

**IP Port**                 Each Host IP Name has an associated IP Port number which you can change
                            using this field.  Normally TN3270 is defined to port 23 but in some cases your
                            connection may need to use a different port number. This port number has to be
                            the same as that specified for the CNSLPORT system parameter in the Hercules
                            Configuration File.

                            If you need to logon to the same hostname using various port numbers create
                            multiple Alias names which point to the same host but use different ports.
                            Additionally you can define aliases for IP addresses. This can help you remem-
                            ber which host Vista is connecting to.

**Delete Host**             This button can be used to delete the selected Host IP Name if you want to clean
                            up the list.

**Terminal Model**        Vista can emulate 5 standard terminal models:

                                      Mod 2   (24 lines by 80 columns)

                                        Mod 3   (32 lines by 80 columns)

                                        Mod 4   (43 lines by 80 columns)

                                        Mod 5   (27 lines by 132 columns)

                                        User    (variable up to 72 lines by 200 columns)

**Assign Host to Alias**  Dot addresses and DNS IP names can be cryptic to look at. Instead you can type a more descriptive name such as "P390" in the Host IP Name or Alias field then press the Assign Host to Alias button to relate the alias name to a real DNS name or dot address. Alias names can also provide the ability to logon to the same hostname using different port or LU names.

**TN3270E**              TN3270E is the default protocol for Vista connections.

**LU Name**              When in TN3270E mode, you have the option of specifying a VTAM 8 character LUNAME to be used when establishing the connection. If you need to logon to the same hostname using various LU names, create multiple Alias names which point to the same host but have different LU's.

**Connect Button**       When this button is pressed, Vista attempts to connect to the specified Host IP. If all is correct Vista connects to Hercules and presents the Hercules welcome screen. If however there is a problem you may see an error window such as the following one:
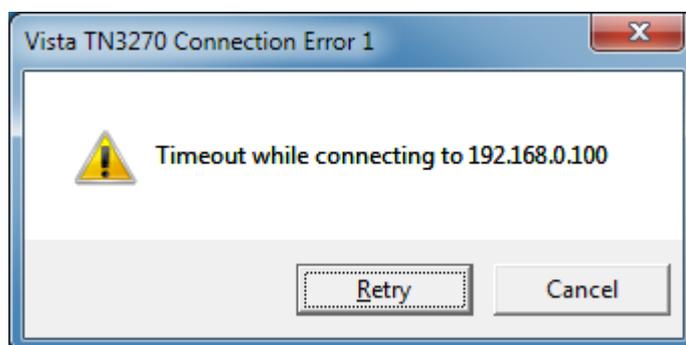


**Figure 79: Vista tn3270 - Connection Error**

# 13. Installation of XMIT Manager



**Figure 80: XMIT Manager Logo**

## 13.1 XMIT Manager Basics

The XMIT Manager is a Windows based application that allows you to manipulate Xmit format files gene-rated on an IBM mainframe. With XMIT Manager you can open Xmit files and view or extract both, binary or text data contained within them. Xmit Manager will process both partitioned and sequential datasets using a graphical interface.

## 13.2 Downloading the Binaries

The binaries for the XMIT manager can be downloaded from various locations. The most reliable and therefore recommended source is the CBT website which can be accessed with the following link:

http://www.cbttape.org/njw/

## 13.3 Installation Steps

Installation of the XMIT Manager is straight forward. First extract the ZIP-file you downloaded from the CBT website to a directory on your harddisk. This creates some files after which you start the installation by running the SETUP.EXE program.

The setup program first presents a screen showing the Software License Agreement.
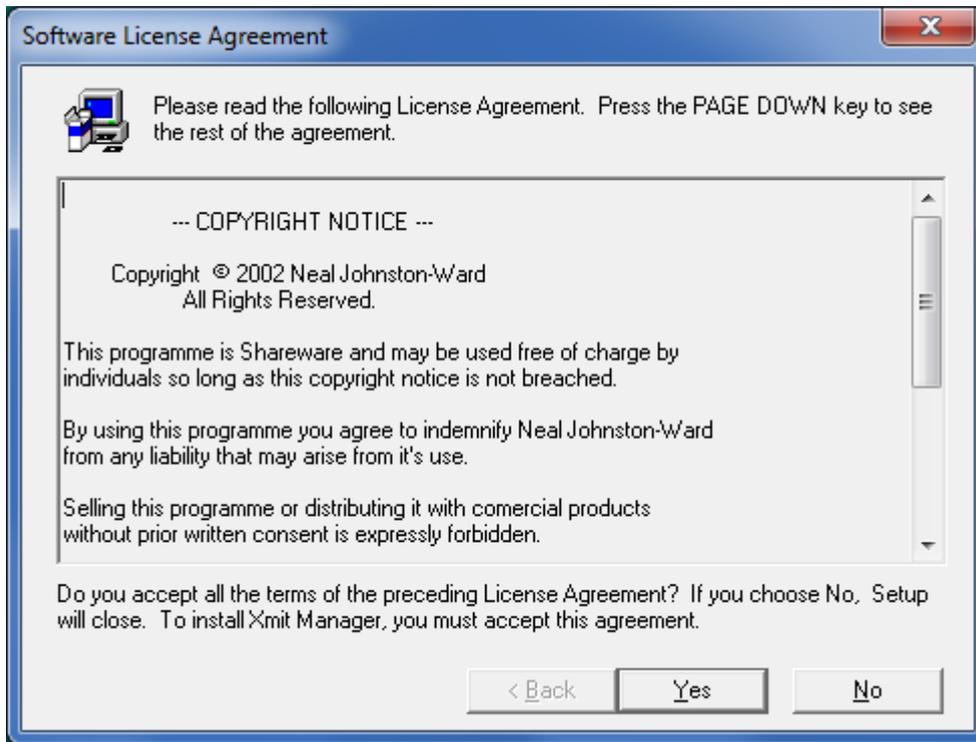


**Figure 81: XMIT Manager Setup – Software License Agreement**

Read the "Software License Agreement" and accept it by clicking on "Yes".

The destination location screen will be shown, from here choose the directory where you want the setup program to install the software. The default location "C:\Program Files\Xmit Manager" is usually satisfactory. If you wish to install the software somewhere else click on "Browse" and chose your preferred drive and directory, or type a path directly in the panel.
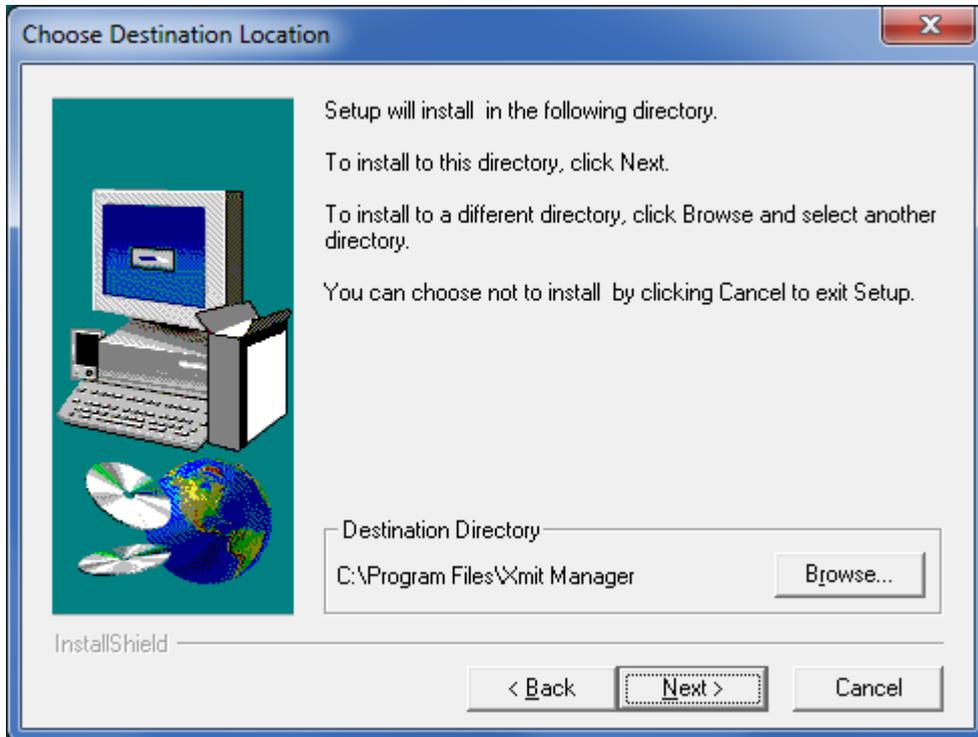


**Figure 82: XMIT Manager Setup – Destination Location**

When you are satisfied with your choice of the installation directory click on "Next >" to continue.

The setup program will ask you to select a program folder, the default is normally acceptable.
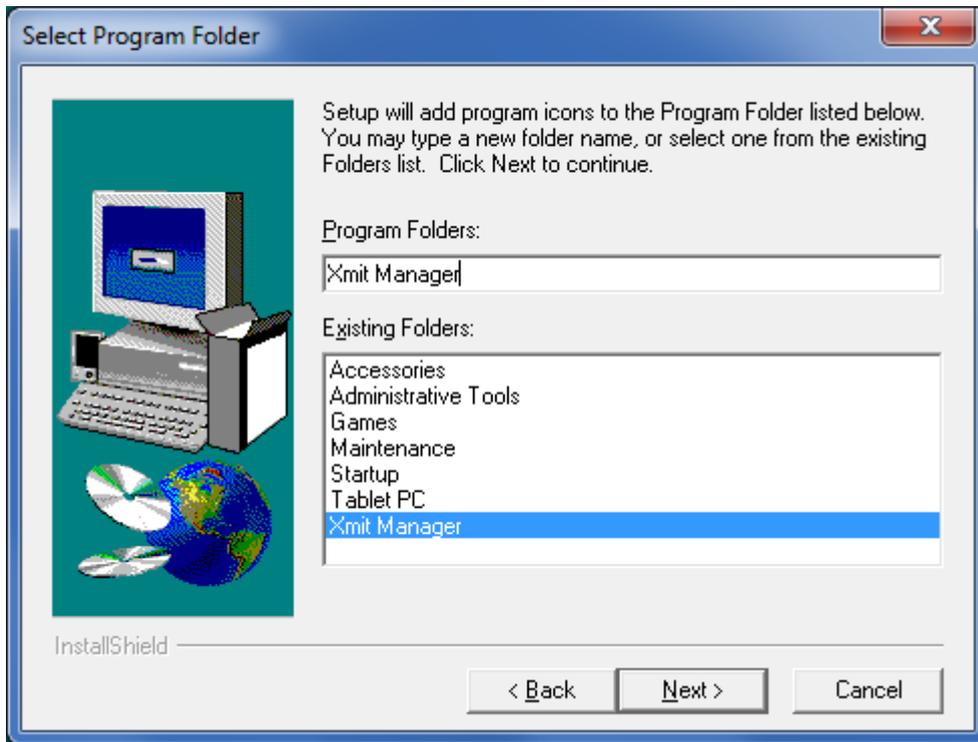


**Figure 83: XMIT Manager Setup – Select Program Folder**

Click on "Next >" again to proceed with the installation.

The following screen gives you a chance to review your selections and to go back and correct any, if necessary.
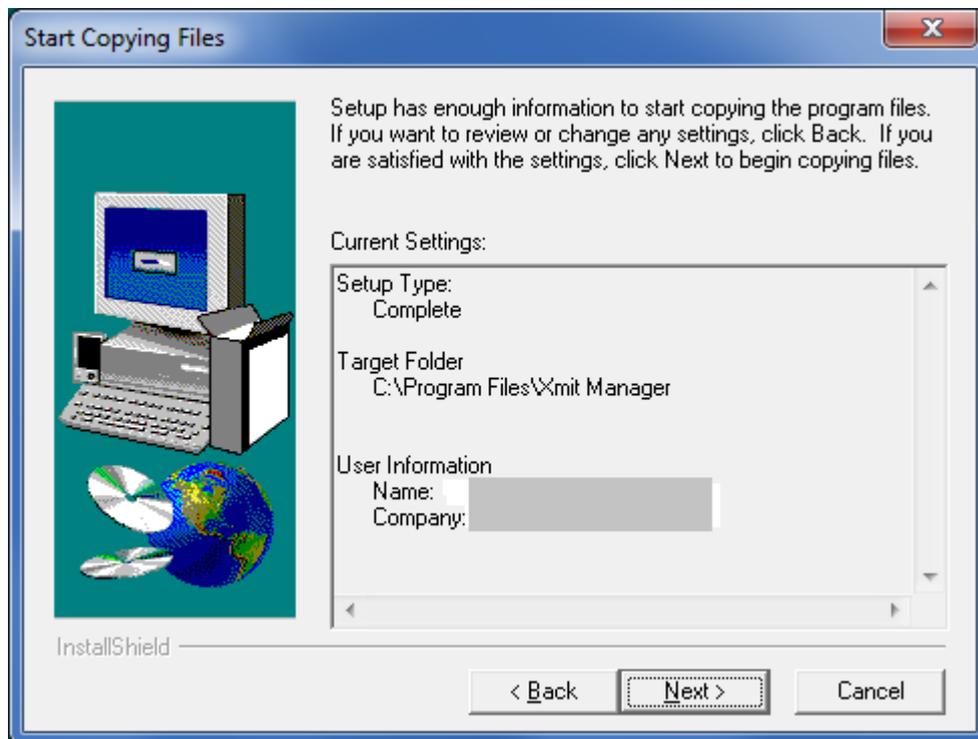


**Figure 84: XMIT Manager Setup – Review Settings**

When you are satisfied with your installation choices, continue by clicking "Next". The installer will start copying files to your harddisk, creating icons and updating registry settings.

The final setup screen will then be displayed.
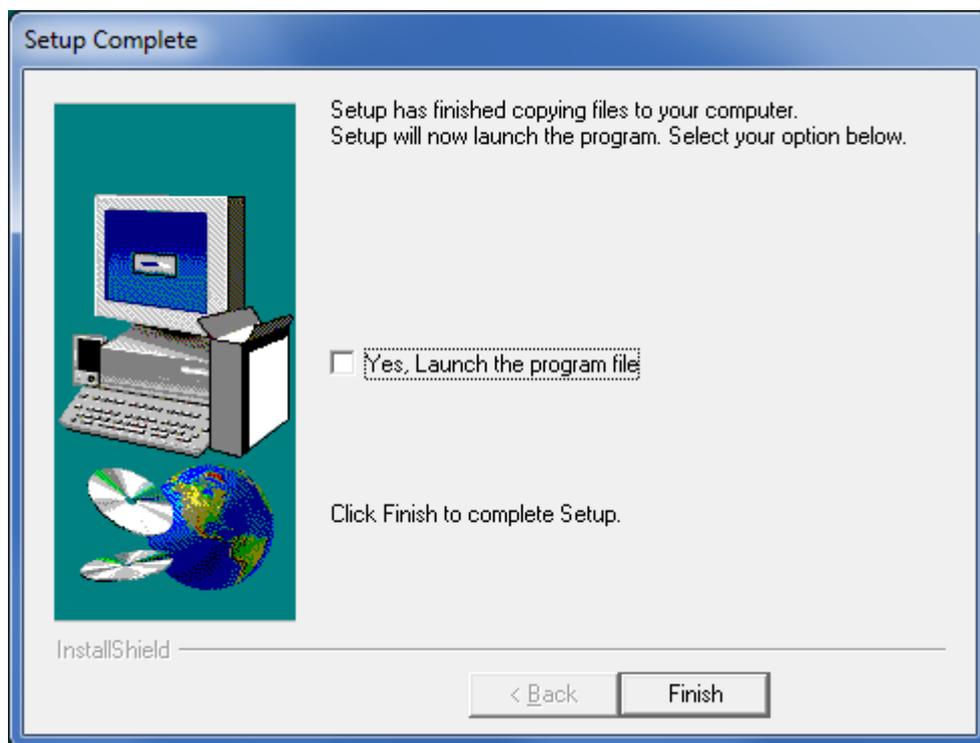


**Figure 85: XMIT Manager Setup – Setup Complete**

You can now either exit the setup program or optionally select to launch the application as you exit setup. Click on "Finish" to complete.

# 14. AWS Browse

## 14.1 AWS Browse Basics

The AWS Browse Utility can be used to view the contents of emulated mainframe tapes (AWS and HET) directly from the Windows desktop without having to start a mainframe operating system. There are currently two implementation of AWS browse.

The first and older one from Rob Storey is no longer supported. The second one is from David B. Trout (Fish), which has more features and is the subject of the rest of this chapter.

## 14.2 Downloading the Binaries

The binaries for the AWS Browse utility can be downloaded from various locations. However the recommended source is the developers to ensure you have the latest version. The following link leads you directly to the download page:

http://www.softdevlabs.com/Hercules/hercgui-index.html

**Note:** Beginning with release 1.5.1.1805 of AWS Browse additional DLLs are required. These are Microsoft MFC and VC Runtime DLLs that you can download from the address mentioned above. The installation takes a few seconds and does not require a reboot. There are a 32-bit and a 64-bit version of these DLLs. Please ensure you are using the correct one according to the product you are installing (32-bit or 64-bit version of the Windows GUI).

If you previously installed the Hercules Windows GUI as described earlier in this manual then these DLLs are already present on your system. Note that you only need to install these C Runtime DLLs once even if new versions of AWS Browse are subsequently installed.

## 14.3 Installation Steps

Download the file called "AWSBrowse-*version*.zip" to your harddisk. 'Version' is the version number of the utility in the form "v.r.m.b" (version.release.modification.build), i.e. *AWSBrowse-1.2.0.1278.zip.*

The archive contains several files; the actual AWSBrowse EXE files (32-bit and 64-bit versions) and some DLLs. Please check the README file for the latest instructions.

Unzip these files and copy them to either your Hercules directory (see Chapter 8. Hercules Emulator Installation) or a directory of your choice from where you wish to run the utility. Be sure to you place the executable and the DLLs in the same directory. The Installation is now complete.

Clicking on the desired executable starts AWS Browse with the initial screen showed below.
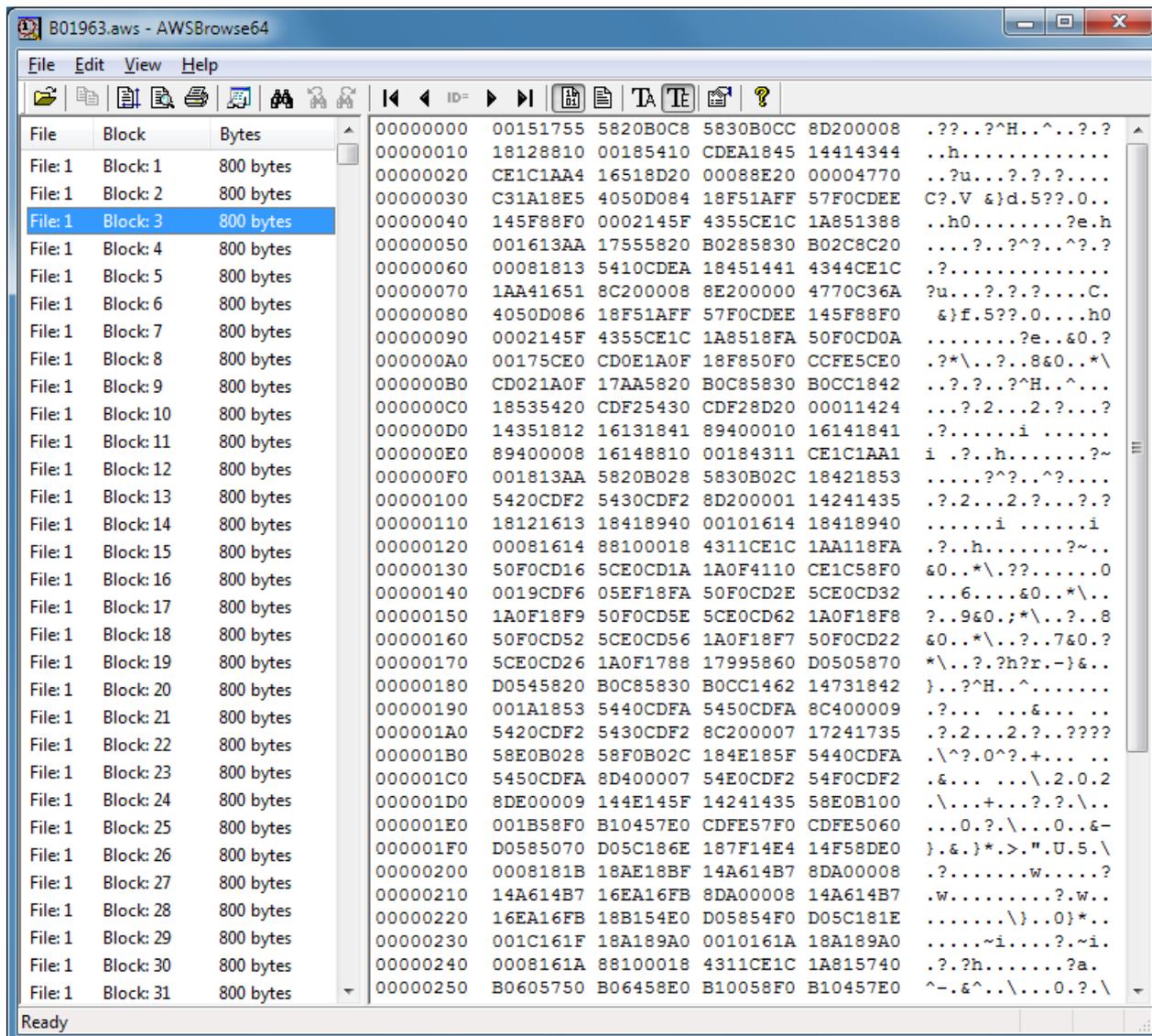
**Figure 86: AWS Browse - Initial Screen**

# 15. Hercules "MSVC" Build Instructions

## 15.1 Introduction

This section provides instructions on how to build the MSVC version of Hercules. Once you have successfully set up the build environment (which you only need to do once) the process of actually building Hercules is very simple. Just open the provided 'Hercules.sln' solution file and click the 'Build' or 'Rebuild All' button.

## 15.2 Setting up the Hercules build environment

The overall setup and build process consists of the following steps:

1. Download and install Visual C++ 2008 Express
   (http://www.microsoft.com/express/download/#webInstall).

2. Install the ZLIB package. (optional, details see below)

   a. Download the package
      (http://www.softdevlabs.com/Hercules/ZLIB1-1.2.3-bin-lib-inc-vc2008-x86-x64.zip).

   b. Unzip to a directory of your choice.

   c. Define a ZLIB_DIR environment variable pointing to the path from the previous step.

3. Install the BZIP2 package (optional, details see below).

   a. Download the package
      (http://www.softdevlabs.com/Hercules/BZIP2-1.0.5-bin-lib-inc-vc2008-x86-x64.zip).

   b. Unzip to a directory of your choice.

   c. Define a BZIP2_DIR environment variable pointing to the path from the previous step.

4. Install the PCRE package (optional, details see below).

   a. Download the package
      (http://www.softdevlabs.com/Hercules/PCRE-6.4.1-bin-lib-inc-vc2008-x86-x64.zip).

   b. Unzip to a directory of your choice.

   c. Define a PCRE_DIR environment variable pointing to the path from the previous step.

5. Start Visual C++ 2008 Express, open the Hercules provided "Hercules.sln" solution file and click on the 'Build' or 'Rebuild All' button.

## 15.3  Setting up ZLIB Support

ZLIB is a compression algorithm written by Jean Loup Gailly and Mark Adler and may be used in the Hercules project pursuant to the ZLIB license.

In source form the Hercules project does not contain any ZLIB source code at all. In binary form however the Hercules project may include an unmodified version of the ZLIB runtime DLL in addition to its own distribution binaries.

The ZLIB_DIR environment variable defines the location where the files required for building a version of Hercules with ZLIB compression support are found. The makefile used by the Hercules build process tests whether this environment variable is defined and acts accordingly to build Hercules with or without ZLIB compression support.

If ZLIB_DIR is undefined when building Hercules then an attempt is made to locate the ZLIB library in a predefined default directory. If it does not find it then ZLIB support will not be generated. Otherwise ZLIB_DIR must point to a valid directory where the ZLIB package is installed.

ZLIB_DIR should contain the path of the ZLIB directory, where the following file / directory layout is expected:

```
\$(ZLIB_DIR)
    zlib1.dll
    zlib1.pdb
    \Debug
    \include
        zconf.h
        zlib.h
    \lib
        zdll.lib
    \x64
        zlib1.dll
        zlib1.pdb
        \Debug
        \include
            zconf.h
            zlib.h
        \lib
            zdll.lib
```

**Figure 87: ZLIB directory layout**

When building a 64-bit (x64) version of Hercules the above 'x64' subdirectories are automatically searched, so as long as the above directory structure is adhered to then Hercules should build fine.

## 15.4  Setting up BZIP2 Support

BZIP2 is a freely available open source, BSD-style license, patent free, high-quality data compressor written by Julian R. Seward. It typically compresses files to within 10% to 15% of the best available techniques from the PPM family of statistical compressors, and performs approximately twice as fast at compression and six times faster at decompression than comparable implementations.

In source form the Hercules project does not contain any BZIP2 source code at all. In binary form however the Hercules project may include an unmodified version of the BZIP2 runtime DLL in addition to its own distribution binaries.

The BZIP2_DIR environment variable defines the location where the files required for building a version of Hercules with BZIP2 compression support are found. The makefile used by the Hercules build process tests whether this environment variable is defined and acts accordingly to build Hercules with or without BZIP2 compression support.

If BZIP2_DIR is undefined when building Hercules then an attempt is made to locate the BZIP2 library in a predefined default directory. If it does not find it then BZIP2 support will not be generated. Otherwise BZIP2_DIR must point to a valid directory where the BZIP2 package is installed.

BZIP2_DIR should contain the path of the BZIP2 directory, where the following file / directory layout is expected:

```
\$(BZIP2_DIR)
    \Debug
    bzlib.h
    libbz2.dll
    libbz2.lib
    libbz2.pdb
    \x64
        \Debug
        bzlib.h
        libbz2.dll
        libbz2.lib
        libbz2.pdb
```

**Figure 88: BZIP2 directory layout**

When building a 64-bit (x64) version of Hercules the above 'x64' subdirectories are automatically searched, so as long as the above directory structure is adhered to then Hercules should build fine.

## 15.5  Setting up PCRE Support

PCRE (Perl-Compatible Regular Expressions) is a set of functions that implement regular expression pattern matching using the same syntax and semantics as Perl 5. PCRE has its own native API as well as a set of wrapper functions that correspond to the POSIX regular expression API. The PCRE library is free, including use in commercial software.

In source form the Hercules project does not contain any PCRE source code at all. In binary form however, the Hercules project may include an unmodified version of the PCRE runtime DLLs in addition to its own distribution libraries.

The Perl-Compatible Regular Expressions library is needed only to support the Hercules Automatic Operator (HAO) Facility. If you do not plan to use the Hercules Automatic Operator Facility then you do not need to install PCRE support and you may skip this step.

The PCRE_DIR environment variable defines the location where the files required for building a version of Hercules with PCRE support are found. The makefile used by the Hercules build process tests whether this environment variable is defined and acts accordingly to build Hercules with or without PCRE support.

If PCRE_DIR is undefined when building Hercules then an attempt is made to locate the PCRE library in a predefined default directory. If it does not find it the regular expression support will not be generated. Otherwise PCRE_DIR must point to a valid directory where the PCRE package is installed.

PCRE_DIR should contain the path of the PCRE directory, where the following file / directory layout is expected:

```
\$(PCRE_DIR)
   \bin
      pcre3.dll
      pcre3.pdb
      pcreposix3.dll
      pcreposix.pdb
   \include
      pcre.h
      pcreposix.h
   \lib
      pcre.lib
      pcreposix.lib
   \x64
      \bin
         pcre3.dll
         pcre3.pdb
         pcreposix3.dll
         pcreposix.pdb
      \include
         pcre.h
         pcreposix.h
      \lib
         pcre.lib
         pcreposix.lib
```

**Figure 89: PCRE directory layout**

When building a 64-bit (x64) version of Hercules the above 'x64' subdirectories are automatically searched, so as long as the above directory structure is adhered to then Hercules should build fine.

## 15.6  Building Hercules using the Visual Studio IDE

Default Visual Studio 9.0 (Visual C++ 2008 Express) solution and project files are included as part of the Hercules source-code distribution.

To build Hercules just open the 'Hercules.sln' solution file and click the 'Build' or 'Rebuild All' button. When you click the 'Rebuild All' button the makefile project simply invokes 'makefile.bat' which in turn invokes the 'nmake' command for 'makefile.msvc' after calling few batch files to define the MSVC build environment.

# Part III: Linux Installation

# 16. Software Prerequisites

## 16.1 Operating System

Hercules is an open source software implementation of the mainframe System/370, ESA/390 and z/Architecture hardware. Hercules itself is not an operating system nor does it emulate a mainframe operating system. The Hercules Emulator runs under Linux on several hardware platforms including the Intel Pentium PC, under various flavors of Microsoft Windows and under MAC OS X. From the point of view of the underlying operating system the Hercules Emulator is just an application program.

Part III of this guide focuses solely on the installation of Hercules under Linux. Details of other host operating systems are covered in Part II (Windows) and Part IV (Mac OS X) in this book. The installation of any hosted operating system and software utilities is beyond the scope of this book.

### 16.1.1 Linux Variants

As is commonly known, there are many variants of the Linux operating system started by Linus Torvalds in 1991. For the purposes of this document, we will group the variants together based on the distribution they are derived from. The impact of the differences between the distributions as it affects the Hercules installation and configuration process lies primarily with the tools used to perform the various tasks and not with the operating systems capability of running the emulator.

The installation and configuration of Hercules on Linux described in this manual is performed using an Ubuntu 10.04 ("Lucid Lynx"). This is a Debian based distribution and the details described in the following chapters should apply equally to most Debian based variants. Where a difference exists between the processed used for the Debian based installation and that used for another distribution, an indication will be made showing the alternate process to be used for the annotated distribution.

The distributions identified in this document should not be seen as a limitation or restriction to the Linux variants on which Hercules can be installed. The limitation applies only to those variants tested by the author in creating this manual.

### 16.1.2 Stability

Whilst most Linux systems are generally less cluttered than their Window counterparts, the stability of the system running Hercules will very much depend on what other processes will co-exist within the same system.

If you are using a Linux environment dedicated to only run the Hercules emulator, whether as a virtual machine or a Linux server, then the stability of the system should be of little concern. If however, you use a Linux system that is also your primary desktop environment, then, as with Windows desktop systems, the more software that is installed and runs parallel to Hercules the greater the chances of the stability being impacted.

### 16.1.3 Installed Software on a Hercules System

It was recommended earlier to have only a minimum of software installed on a Hercules host machine. Whilst this is a good rule of thumb, Linux is pretty good at running multiple workloads if you have the hardware resources to allow it to do this effectively.

However, to make your own life easier, the following software recommendations should be borne in mind:

- Linux variants based on the 2.4.0 kernel (or later) released in 2001 are the simplest to configure for use with Hercules however it is possible to use kernels prior to this

- Software firewall, especially if there is no hardware firewall in the LAN

- Antivirus software with on demand and on access checks active

Depending on your requirements other commonly used utilities include:

- Network sniffer

- Performance monitor / Task monitor

- FTP program

In general, the fewer software applications that are running on the system, the better the emulated main-frame will run.

## 16.2 Runtime Environments

The Linux 2.4.0 kernel (or later) contains all the components required to run Hercules. Additional software packages may be required in order to run extended functionality of the Hercules software stack (such as Hercules Studio from Jacob Dekel), but no changes should be required to the kernel itself.

## 16.3 Runtime Security

The Hercules Emulator can run under a root userid (any userid with a UID of 0), but it is recommended to run it under its own userid and group. To do this, it is recommended to create a new group for the Hercules files and directories and a userid under which to run the Hercules Emulator itself.

## 16.4 Hercules Emulator

The Hercules Emulator consists of the following mandatory and optional components:

- Hercules Emulator (mandatory)

- Hercules Studio GUI (optional)

- Hebe - Hercules Image Manager (optional)

- Additional utilities (optional)

### 16.4.1 Hercules Binaries

The Hercules executables are the heart of the emulator and a mandatory component. This is the software implementation of the System/370, ESA/390 and z/Architecture mainframe hardware and processor machine code instruction set.

Hercules runs as a shell program and comes with a semi-graphical display in a shell terminal (the Hercules Hardware Console - HMC) consisting of two screens, switched between using the ESC key. When the Hercules HTTP server is running then Hercules can also be operated via a web browser.
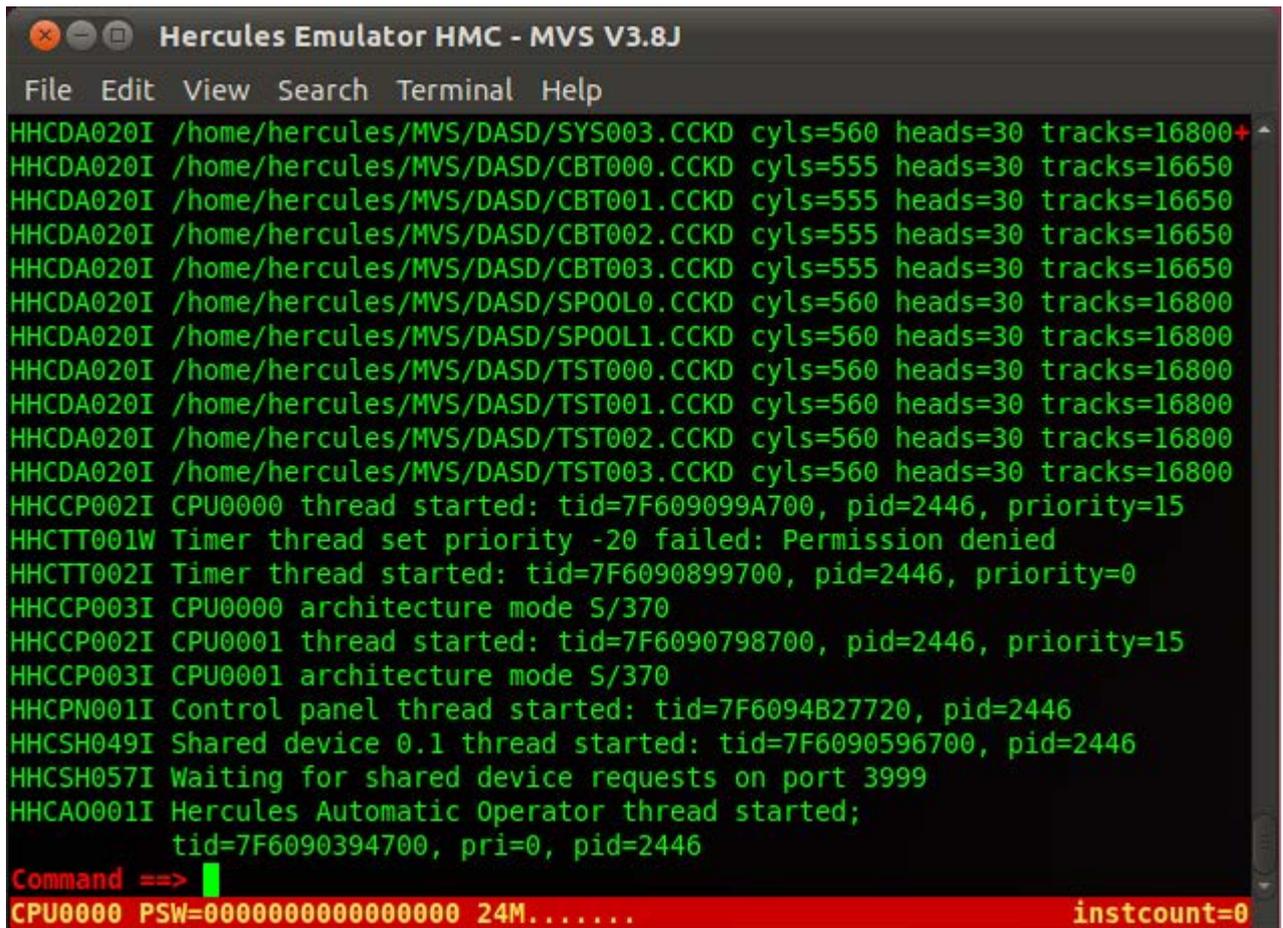
```
🔴🟠🟢  Hercules Emulator HMC - MVS V3.8J
File  Edit  View  Search  Terminal  Help
HHCDA020I /home/hercules/MVS/DASD/SYS003.CCKD cyls=560 heads=30 tracks=16800+
HHCDA020I /home/hercules/MVS/DASD/CBT000.CCKD cyls=555 heads=30 tracks=16650
HHCDA020I /home/hercules/MVS/DASD/CBT001.CCKD cyls=555 heads=30 tracks=16650
HHCDA020I /home/hercules/MVS/DASD/CBT002.CCKD cyls=555 heads=30 tracks=16650
HHCDA020I /home/hercules/MVS/DASD/CBT003.CCKD cyls=555 heads=30 tracks=16650
HHCDA020I /home/hercules/MVS/DASD/SPOOL0.CCKD cyls=560 heads=30 tracks=16800
HHCDA020I /home/hercules/MVS/DASD/SPOOL1.CCKD cyls=560 heads=30 tracks=16800
HHCDA020I /home/hercules/MVS/DASD/TST000.CCKD cyls=560 heads=30 tracks=16800
HHCDA020I /home/hercules/MVS/DASD/TST001.CCKD cyls=560 heads=30 tracks=16800
HHCDA020I /home/hercules/MVS/DASD/TST002.CCKD cyls=560 heads=30 tracks=16800
HHCDA020I /home/hercules/MVS/DASD/TST003.CCKD cyls=560 heads=30 tracks=16800
HHCCP002I CPU0000 thread started: tid=7F609099A700, pid=2446, priority=15
HHCTT001W Timer thread set priority -20 failed: Permission denied
HHCTT002I Timer thread started: tid=7F6090899700, pid=2446, priority=0
HHCCP003I CPU0000 architecture mode S/370
HHCCP002I CPU0001 thread started: tid=7F6090798700, pid=2446, priority=15
HHCCP003I CPU0001 architecture mode S/370
HHCPN001I Control panel thread started: tid=7F6094B27720, pid=2446
HHCSH049I Shared device 0.1 thread started: tid=7F6090596700, pid=2446
HHCSH057I Waiting for shared device requests on port 3999
HHCAO001I Hercules Automatic Operator thread started;
          tid=7F6090394700, pri=0, pid=2446
Command ==>
CPU0000 PSW=0000000000000000 24M.......                        instcount=0
```

**Figure 90: Hercules Hardware Console – Console Window**

The previous figure shows the initial display in the Hercules console window. The next figure shows the Hercules device and status display accessed with the ESC key.
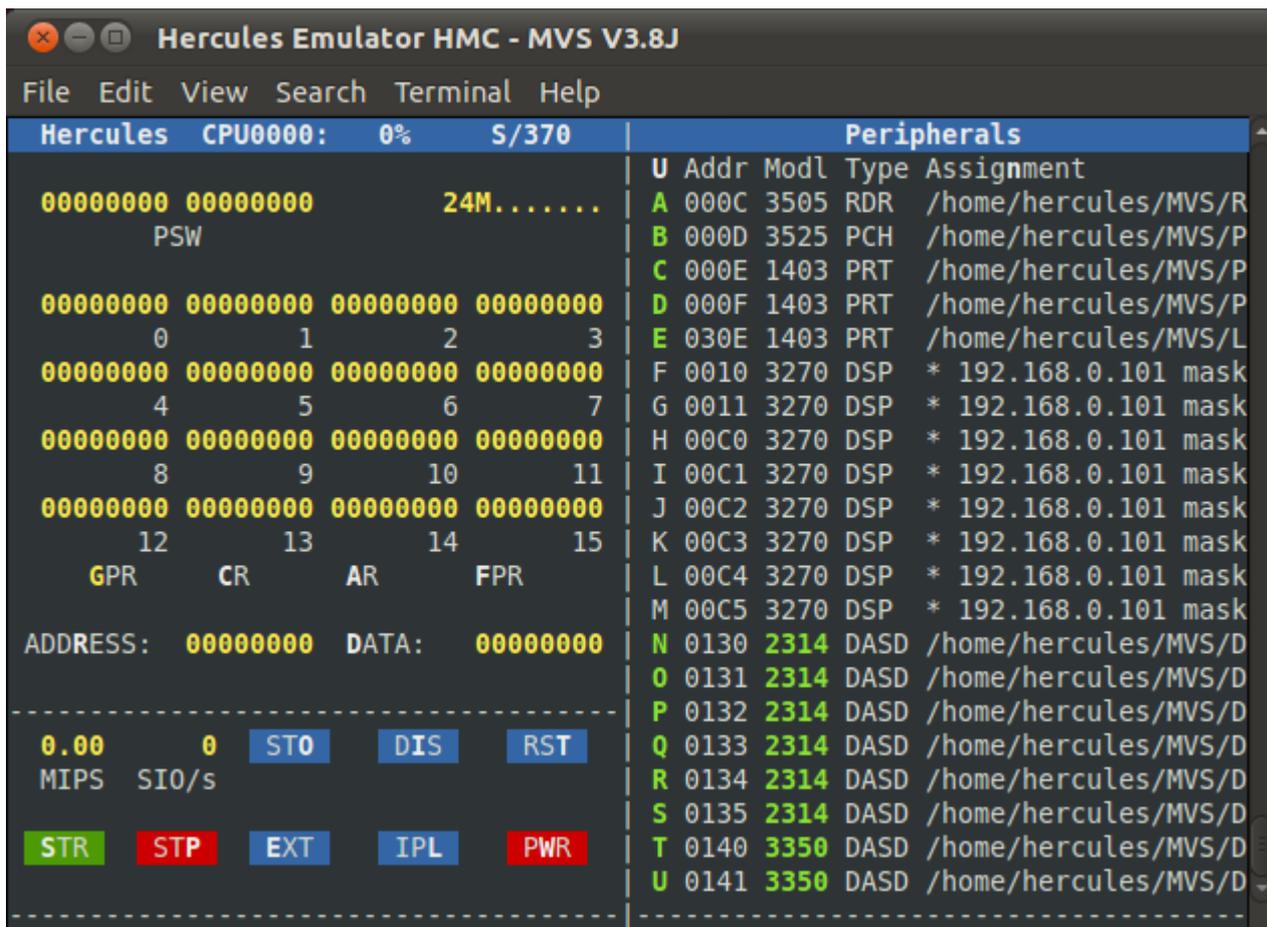
**Figure 91: Hercules Hardware Console – Device and Status Display**

The last figure shows the Hercules web browser interface which can be accessed when the Hercules HTTP server is configured and running.
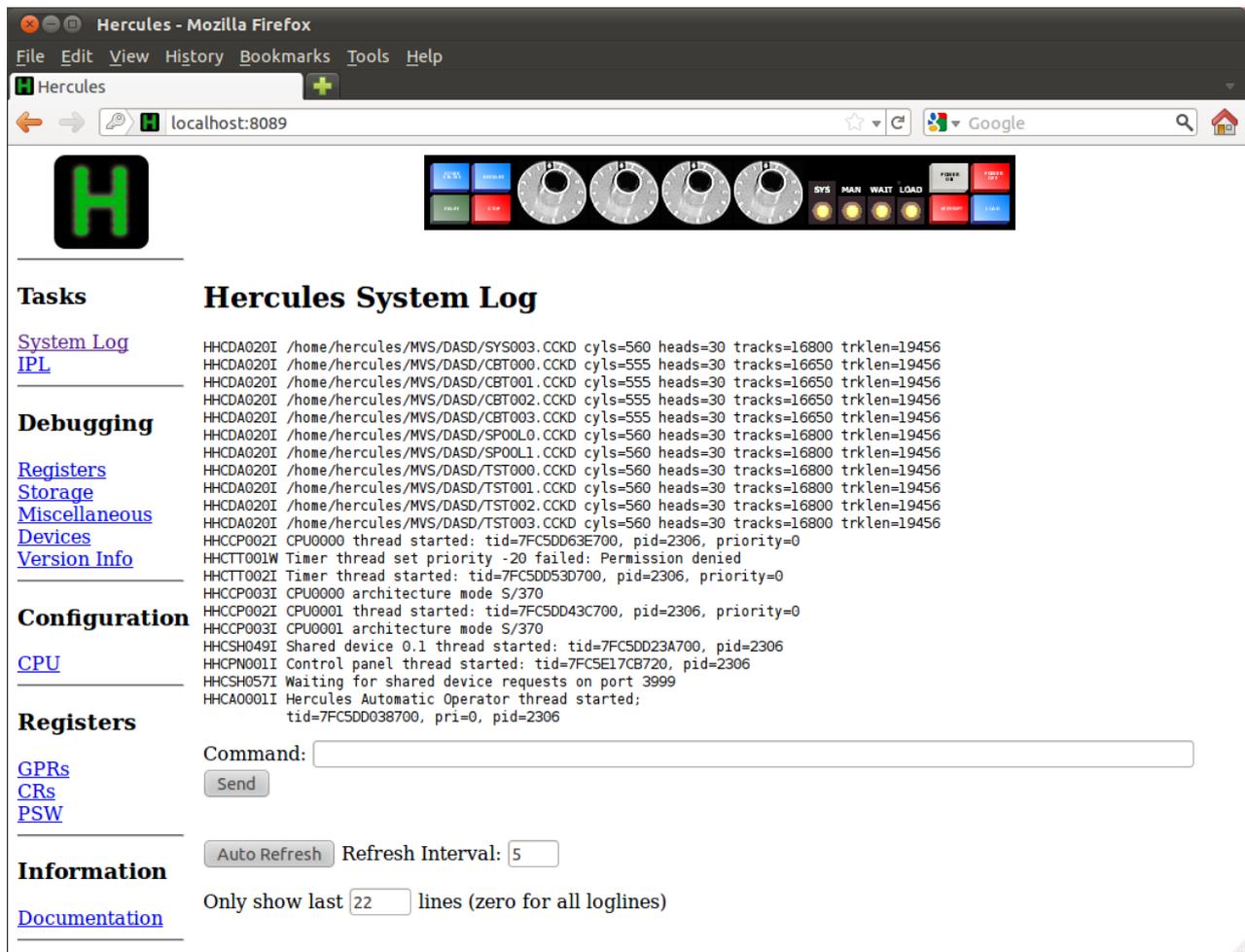
**Figure 92: Hercules Web Browser Interface**

## 16.5 Additional required and optional Software

As well as the components described above other software, either required for practical use of Hercules (e.g. 3270 client) or that makes common tasks easier (e.g. XMIT Manager, AWS Browse, ZZSA etc.) may be used.

### 16.5.1 3270 Client (required)

For virtual 3270 consoles and 3270 terminals a 3270 client software application is required. The 3270 client can run on the same machine as Hercules or on any Linux, Unix or Windows box with a TCP/IP connection to the Hercules machine. One of the most common 3270 clients for Linux and Unix variants is X-windows based x3270.

To install this package (or another similar supported Linux 3270 package), please review the instructions given in "*Software Management*" on page 147.

### 16.5.2  Windows based utilities (optional)

There are a couple of Windows based utilities that can help with the processing of information destined for the IBM mainframe platform.

XMIT Manager is described in "Installation of XMIT Manager" on page 124 and the AWS Browse Utility is described in "AWS Browse" on page130.

# 17. Installing the Hercules Emulator

## 17.1 Installation Preparation

Before we start to install the Hercules binaries and associated software, a word needs to be said about security.

Although installing Hercules in a single-user Linux system may seem to remove the necessity of defining a security environment for it, it is recommended that the security environment is established in order to better show how Hercules relates to the host operating environment.

To create the security environment for Hercules, we will establish a new user and group which will primarily be used for file ownership. The created userid can also be used to run the Hercules process.

### 17.1.1 Hercules Security Group and User

Each variant of Linux has a slightly different implementation of the User and Group management tool.

In our reference Ubuntu system, User and Group management is accessed from: System -> Administration -> Users and Groups. You will need to select a GID and UID for the new security elements. These should be unassigned by any other group or user.



**Figure 93: Selecting Users and Groups**

Once you have started the management tool, navigate to the Group management section or tab and se-lect "*Add*". In Ubuntu, this is the "*Manage Groups*" button.

Enter a name for the Hercules group and enter the GID you have allocated. In this example we have chosen "*hercgrp*" as the group name and a GID of 5000. Click OK to add the group then "*Close*" to end the dialog.
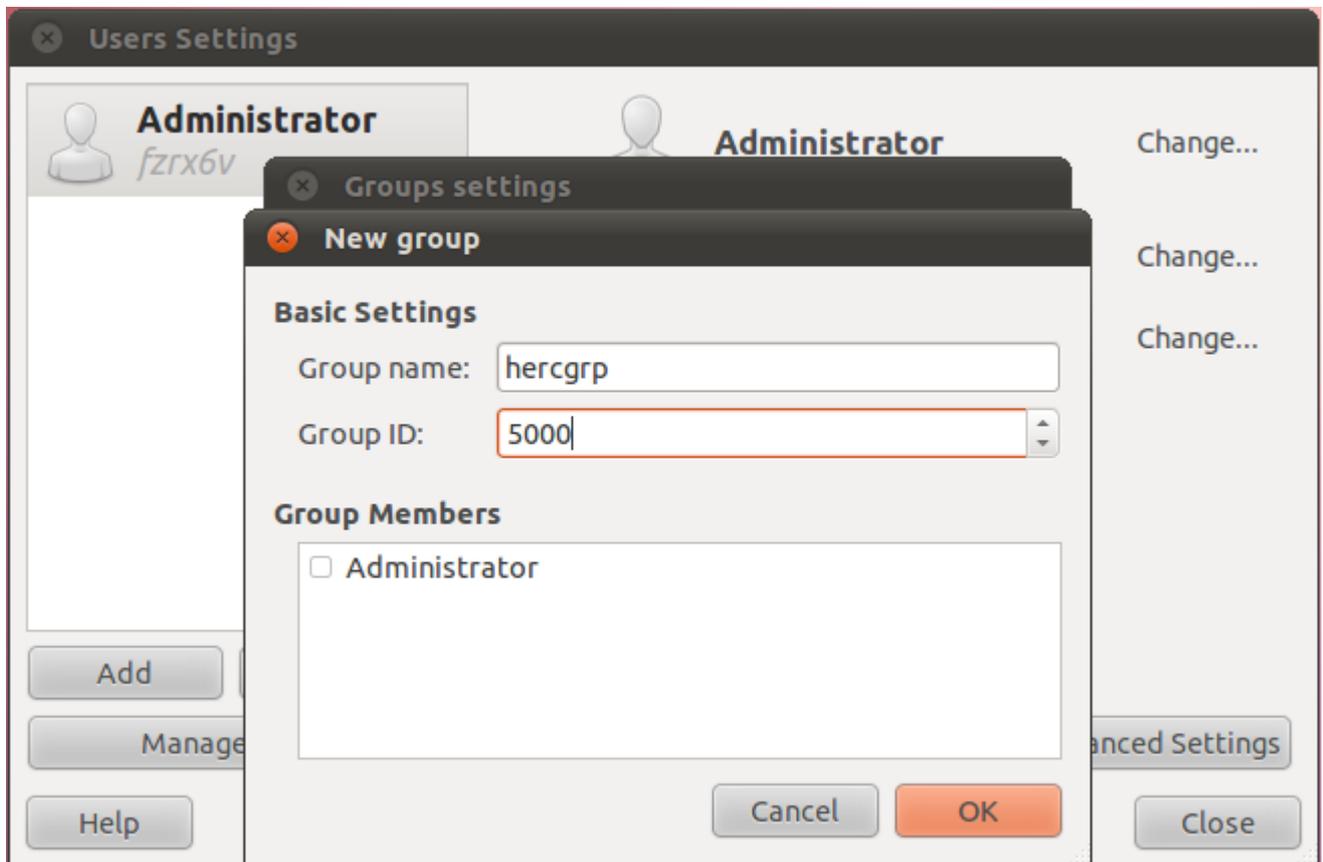


**Figure 94: Creating Linux Group**

Now navigate to the User management section or tab and click "*Add*". In Ubuntu, just click "*Add*" from the "*Users Settings*" dialog.

In the "*Create New User*" dialog, enter the Name and Username of the account to be created and click "*OK*". In this example, we are creating a userid of "hercules".

You will normally be asked to set a password for the new account. Enter your chosen password and click "*OK*" to complete the dialog.
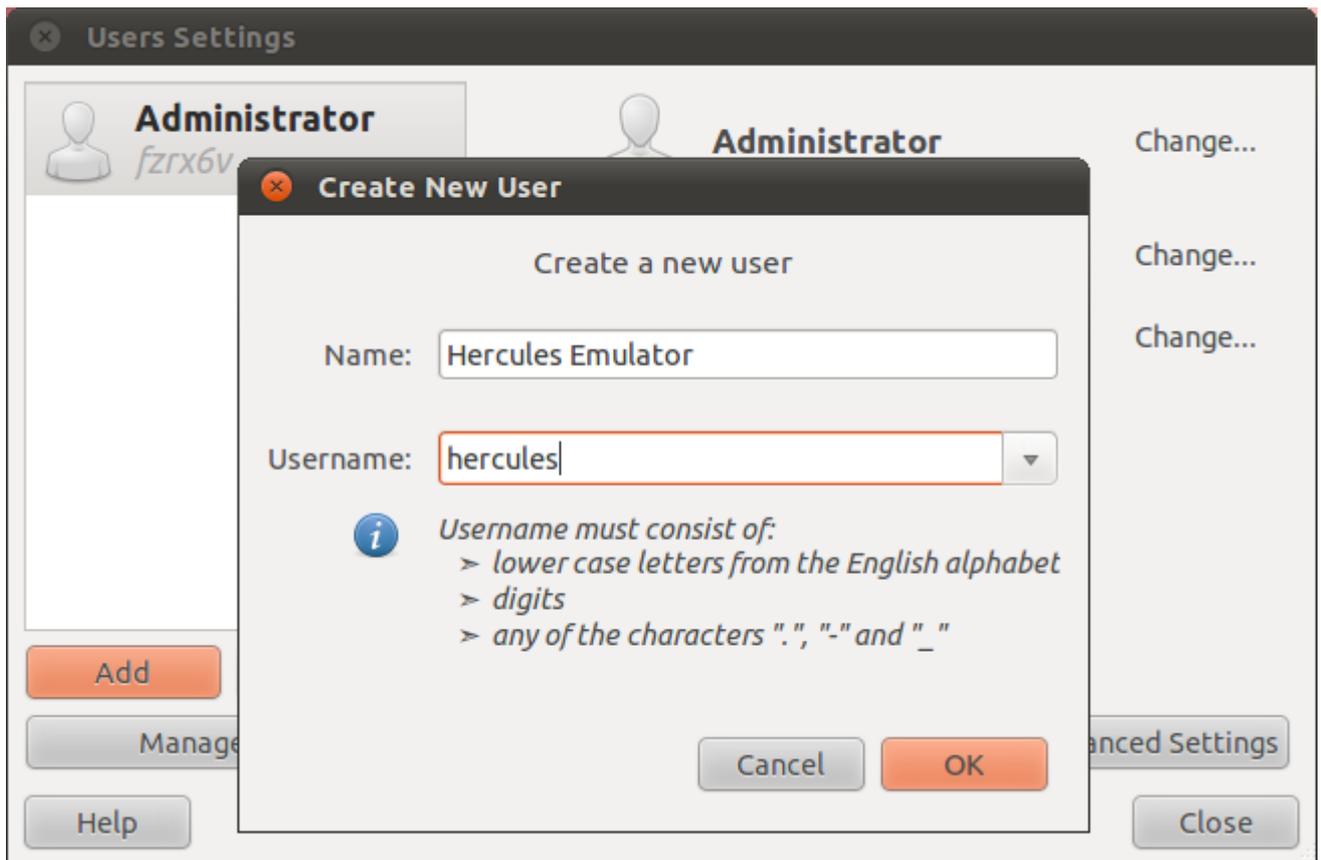
**Figure 95: Create Linux User**

Now click on the "*Advanced Settings*" tab and set the default group to the one you created previously (in this case, "*hercgrp*"). In Ubuntu, this setting is available under the "*Advanced*" tab by selecting the group from the "*Main group*" drop-down list.

If you wish, you can also set a UID for the account. In this example we have set 5000 as the UID (not to be confused with the GID of 5000 for the group).

Click "*OK*" and then "*Close*" to complete the dialog.
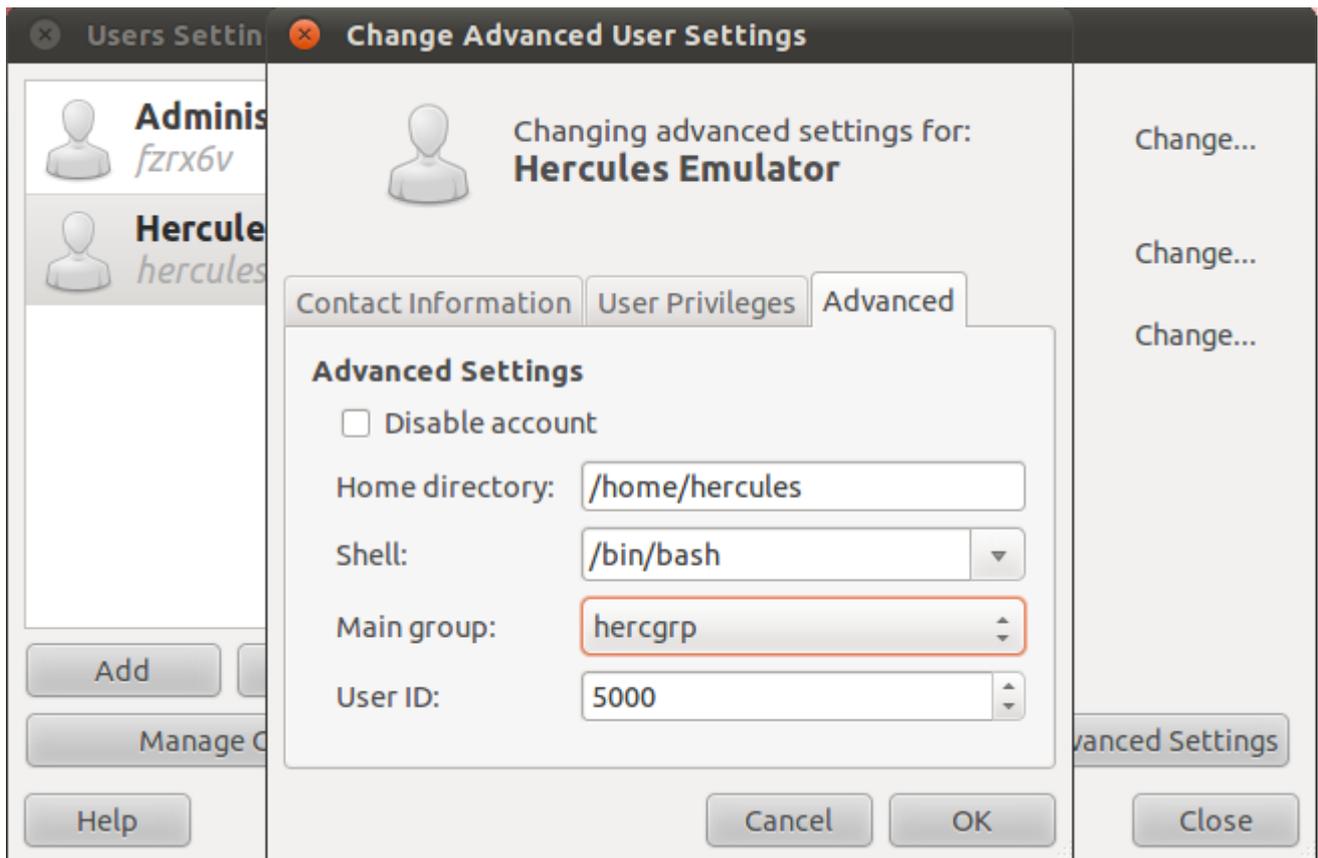
**Figure 96: Advanced User Settings**

## 17.2  Installation Methods

There are a number of ways to install Hercules on Linux and your choice of method may depend both on your flavour of Linux and your attitude to the Hercules software itself.

The different variants of Linux each have their own approach to application software management and many provide excellent tools for installing application packages. If your Linux variant has such a management tool (most do), then this could be the simplest way of installing the Hercules binaries. Our reference Linux system for this document, Ubuntu 10.10, contains such a facility and we will run through the necessary steps to install Hercules using this technique.

If such a plug-and-play approach to software installation does not suite you, or you require greater flexibility when installing software packages, the Hercules web site at http://www.hercules-390.eu contains RPM packages for both binary and source installation.

Both of the above techniques result in the currently released stable version of the Hercules binaries being installed. This is normally the best practice, so either of these two techniques is normally recommended. Some people however, need the latest software being developed by the Hercules engineers and more up-to-date packages can also be implemented.

## 17.3  Software Management Installation

Ubuntu provides a package management facility called the "*Synaptic Package Manager*" which can be found under System -> Administration from the panels bar. To use a program like a Package Management facility, you will normally need to have access to "sudo" and provide your password when you call it.

**Figure 97: Synaptic Package Manager Selection**

The package management software allows you to search for which applications you would like to install.

In this case, enter "hercules" into the quick search box and the software will display a list of applications which contain this name in either the package name or description.

As you can see in the illustration "Package Manager Selection" on page 149, the package manager has responded with a list which contains the Hercules software and a description of "System/370, ESA/390 and z/Architecture Emulator".

By clicking on the check-box to the left of the selection, the system will respond with a properties list detailing the actions that you can perform. In this case, we wish to mark the application for installation which we can do by clicking on this option.
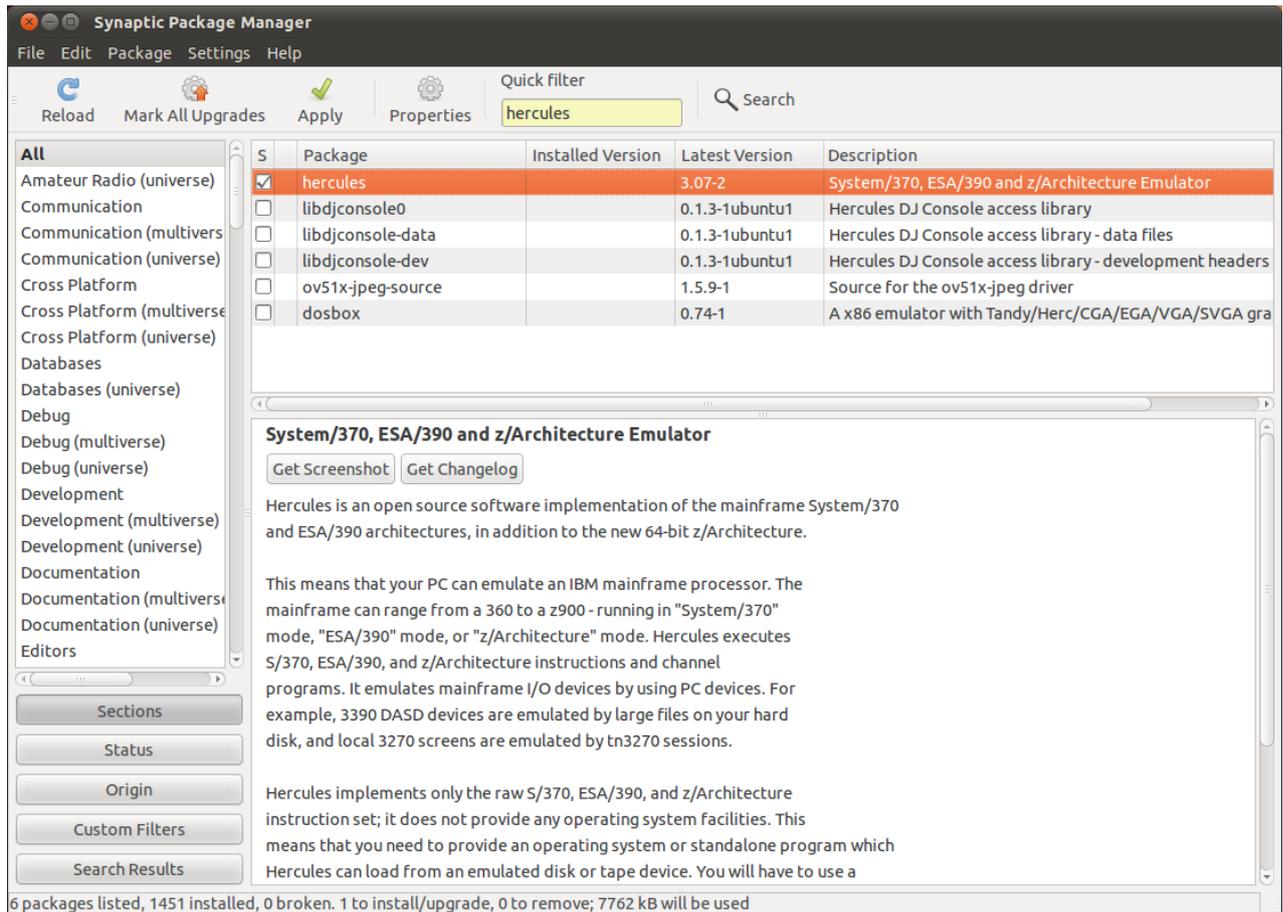
**Figure 98: Package Manager Selection**

Once you have completed your selection of applications to be installed, click the "Apply" icon which you will see on the action bar above the package information. The package manager will respond with a summary showing what will be installed.
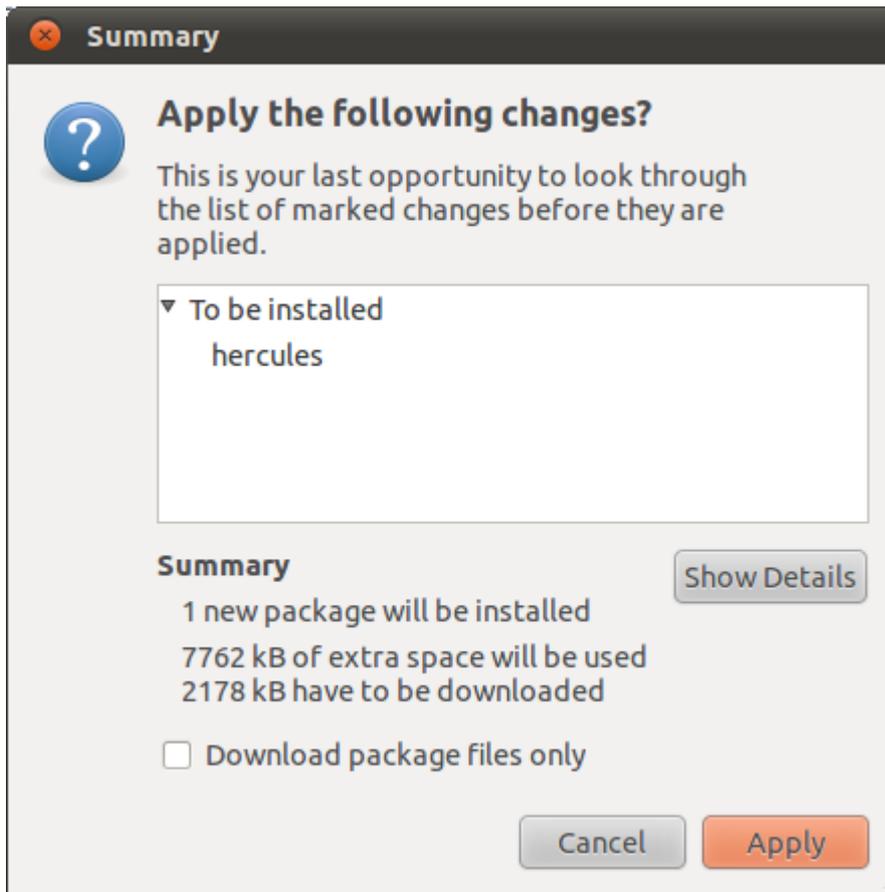
**Figure 99: Applying Application Changes**

Click "*Apply*" on this panel to complete the installation. Once complete, the system will respond with:
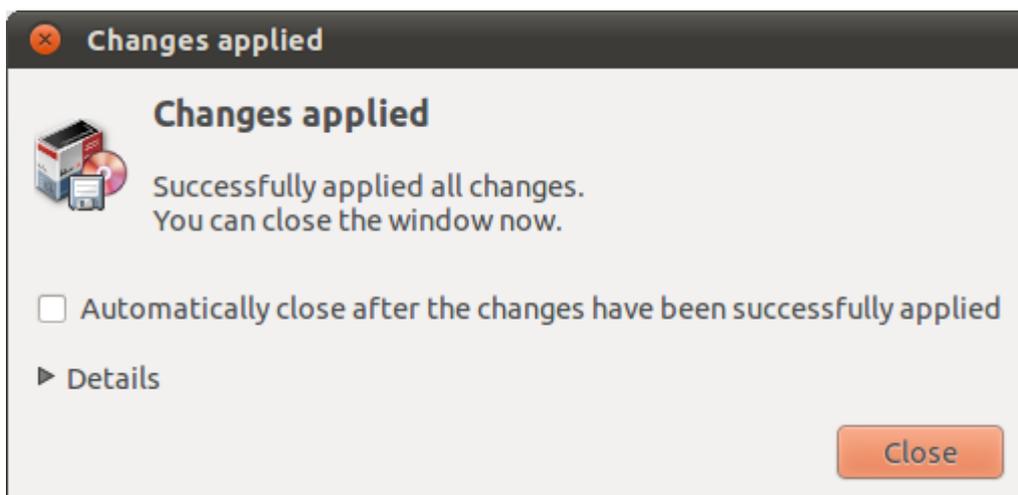


**Figure 100: Changes Applied**

The Hercules binaries are now installed, so you can skip to "Verifying the Hercules Installation" on page 163 to continue.

## 17.4 Building from Source

The second main installation option is to build the Hercules binaries from the available source distribution. Despite the extra effort involved, this method has a significant advantage over the more automated installation options.

The configure and build process allows the installer to specify a number of configuration and optimization flags to control how the binaries are built.

- Configuration options can, for example, enable the external GUI interface that is required for the excellent Hercules Studio offering from Jacob Dekel. They can also provide a mechanism to control where the Hercules binaries are installed. This can be used to easily integrate one of the daily snapshots of the Hercules binaries available from Ivan Warrens indispensable build site.

- Optimization options can build the binaries in such a way that they better exploit the hardware used to run them.

Lastly, this is the only method I suspect a true dyed-in-the-wool system programmer would sanction!

### 17.4.1 The Build Process

We will start the build process by locating and downloading the source package from the Hercules web site. Use one of the following procedures to download the source tarball (Currently described as "hercules-3.07.tar.gz") package to your local software directory (or /tmp if you don't plan to keep it):

1. Point your favorite Browser to http://www.hercules-390.eu and right-click the source tarball link. Then select "Save Link As..." to download it.

2. Use the "wget" command to download the source package:
   - Change to a local software directory

```
cd <local_software_directory>
```

   - Extract the name of the current Hercules source tarball

```
wget -q0 – http://www.hercules-390.eu | egrep "hercules-.*tar.gz"
```

   - Download the file identified by the previous command

```
wget http://www.hercules-390.eu/<identified_file>
```

**Figure 101: Download Hercules Using "wget"**

Now change to a work area (either your home directory or /tmp will do) and run the following command to unpack the downloaded source package:

```
tar -xzvf <local_software_directory>/<identified_file>
```

This will extract all the required files into a sub-directory of your current location which should have the name of the "*identified_file*" (without the suffixes).

**Figure 102: Unpack Binaries using "tar"**

Change into the directory shown after the unpack process is complete. Now issue the following command:

```
./util/cvslvlck
```

This is a handy script provided by the Hercules developers to let us know what we need (that we don't already have) to build the binaries. If any of the software packages that it requires are missing, we will need to use the Software Manager on the Linux system to install them.

The following example output snippet shows one of the packages that the script has identified the following packages as being not present or at a too low level.

**Figure 103: Missing Package**

Open the Package Manager and select all the required packages. As a suggestion add *g++* (the GNU C++ compiler) to this list.

Apply these updates to the system to complete the pre-requisites to configuring and building the binary installation process as shown in the figure below.
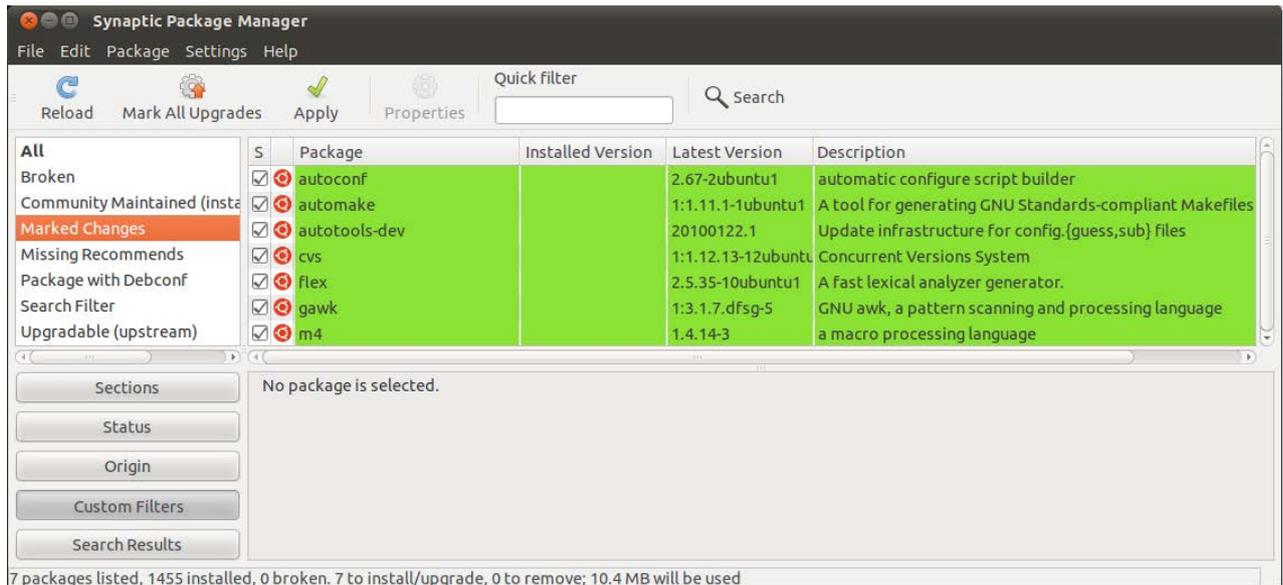
**Figure 104: Installing Required Packages (Package Manager)**

Once the missing packages have been installed, we are ready to configure the installation process. The actual running of this configuration command will vary depending on the hardware you are running Linux on, the variant of Linux you have installed and different packages and their levels. It is the job of the configure process to evaluate all these variables and to create a file that will correctly build the Hercules binaries for your system.

Now we are ready to configure the install process, so we start with the "configure" command. There are many options that can be passed to the configure command, but we will just pass one for this build. Now issue the following command:

```
./configure --prefix=/usr/local/Hercules
```

This option changes the installation target for the binaries. This is useful if we want to later use snapshot versions of the Hercules binaries to test. This parameter is optional and shown here only for illustration. If you chose a new location for the installed binaries, you may want to add the target "bin" and "lib" directories to your PATH and LIBPATH environment variables.

**Figure 105: Configuring the Hercules Install Process**

Once the configure process has completed successfully, we can run the "*make*" command to compile all the source components. Now issue the following command:

```
make
```

The "*make*" command will take some time to complete, but should end without error (warnings should be OK) with messages similar to those shown in this snippet.

**Figure 106: The "make" Command**

The final step of the build process is to link the compiled object files into their binary forms and install them into our target directories.

Now issue the following command:

```
sudo make install
```

We will run the "*make install*" command under root authority so that we can be sure that any required directories can be created. The command should finish without error and the final messages will be similar to those shown in this snippet.

**Figure 107: The "make install" Command**

The Hercules binaries are now installed, so you can skip to "Verifying the Hercules Installation" on page 163 to continue.

## 17.5  RPM Installation

If your flavour of Linux does not have an available version of Hercules to install via its Software Package Manager or you wish to install a version other than the one it offers, you can select to install via the "*RPM Package Manager*" or just RPM. Interestingly, RPM was originally an acronym for "*Red Hat Package Manager*" although it is now referred to as *RPM Package Manager* making the unabbreviated name "*Red Hat Package Manager Package Manager*"?

Some variants of Linux use RPM as their application software management tool. If that is the case, you can skip straight to the "*Download the RPM Package*" section.

For those systems without support for RPM packages, you need to enable support for the local package manager to handle RPM packages or convert the Hercules package to the local package format.

### 17.5.1  Download the RPM Package

Using your favourite browser and navigate to the Hercules main site at http://www.hercules-390.eu. Scroll down to the section describing the available release packages.

Select the Linux RPM that matches your installed Linux software (either 32 or 64 bit). If your Linux variant supports RPM packages natively (such as Fedora), then you will have the possibility to select an install option when downloading the package. When presented with this option, click the "Install" button or select "Open with Package Installer" (depending on your Linux distribution) and skip the rest of this section about RPM installation.
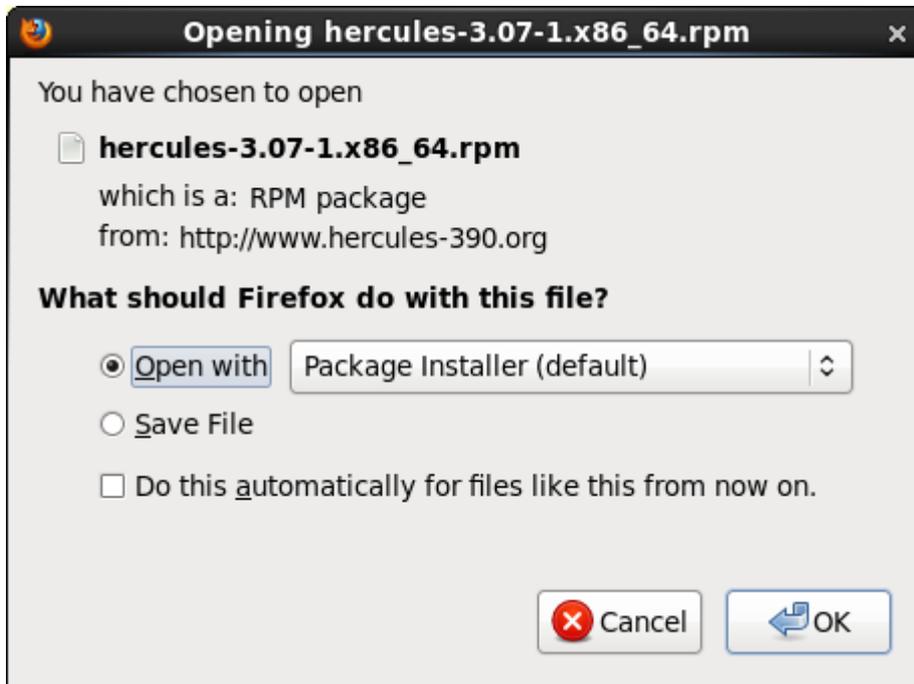


**Figure 108: Install Hercules RPM Package**

If RPM packages are not natively supported (such as Ubuntu) then you will need to right-click on the required RPM package and select "Save Link as" to save the RPM to a local directory and continue with the next section.

## 17.5.2 Installing the RPM Package Manager and Alien

For systems without native RPM support, you will need to install the necessary software to process the RPM package. For Ubuntu, the required software includes the RPM Package Manager and Alien which is a software package conversion tool.

These can be installed using the "*Synaptic Package Manager*" which can be found under "System -> Administration" from the panels bar. To use a program like a Package Management facility, you will normally need to have access to "sudo" and provide your password when you call it.

Start the Package Manager and enter "rpm" into the Quick Search box. This will show you the RPM related packages. Click the "rpm" checkbox and select "Mark for Installation". This will add any other required packages to support the RPM tool.
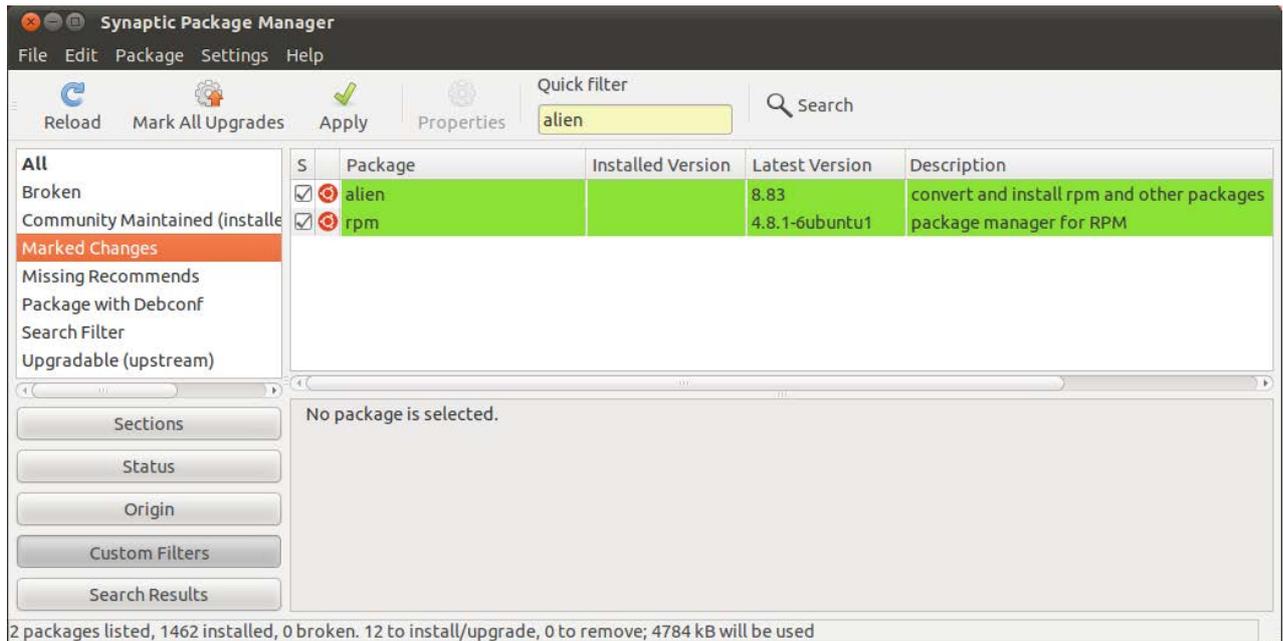
**Figure 109: Install RPM Package Manager**

Similarly, enter "alien" and select the "Mark for Installation". Click "Apply" and wait for the software to be installed. Once complete, the package manager will respond with "Changes applied".

Alternatively, you can use the command interface to install the required software. Open a terminal session and type:

```
sudo apt-get install rpm
sudo apt-get install alien
```

Now continue with "RPM Package Conversion".

## 17.5.3 RPM Package Conversion

For Debian based systems (such as Ubuntu) you will need to convert the downloaded package to a format that is understood by the operating system. The utility "alien" can do this for us but it will leave a prefix string prepended to the installation target directories. If you run the alien conversion and install command, the Hercules package will be installed to "/_prefix/..".

In order to fix this, we need to adjust the RPM package before building the Debian install package from it.

First, we need check the downloaded RPM package to check the supplied version and release. To do this, we use the "*rpm*" command:

```
rpm -qip <rpm-package>
```

This command lists information about the RPM package.



**Figure 110: List RPM Package Information**

Now we need to unpack the RPM package so that we can prepare it for the Debian based system. To do this, we use the "*alien*" command:

```
alien –s <rpm-package>
```

This command unpacks the RPM package into an output directory structure (much like unzip). Change to the unpacked directory and rename the "debian" directory to "DEBIAN".

Next edit the "*conffiles*" file in the DEBIAN directory and replace any "_prefix" string at the beginning of a line with the desired target directory. In this case we will be using the root as our target so we simply remove the "_prefix" strings.

To reflect the changes made in the previous step, move the directory structure and accompanying files under the "_prefix" directory to the target directory structure based on the current directory. As our target directory structure is based on the root directory, we simply move all the files from the "_prefix" directory to the current directory. Once the files have been moved, you can remove the "_prefix" directory.

Next edit the "*control*" file in the DEBIAN directory and add the target version remove the dependencies (they have symbolics which will not be processed correctly).

Now go back to the original directory (back one level) and use the "dpkg" command to build the new Debian installation package:

```
dpkg –b <rpm-dir> <deb-pkg>
```

This command will pack the files from the RPM directory structure into the new Debian format installation package. Assuming all this has gone successfully, we should now have a Debian package ready to install.



**Figure 111: RPM Package Conversion**

In order to simplify the conversion of the RPM to the Debian format package, cut and paste the following into a script file and run it from the directory where the downloaded RPM package file is stored.

```
#!/bin/sh
pkg="${1:-missing_rpm_package}"
echo "*******************************************************************************"
rpm -qip ${pkg} 2> /dev/null || (echo >&2 "rpm failed for package: ${pkg}";exit 1)
```

```
echo "*********************************************************************************"
ver=$(rpm -qip ${pkg} | grep -e "^Version" | tr -s ' ' | cut -d ' ' -f 3)
rel=$(rpm -qip ${pkg} | grep -e "^Release" | tr -s ' ' | cut -d ' ' -f 3)
retline=$(alien -s "${pkg}" 2> /dev/null || (echo >&2 "alien failed for package: ${pkg}";exit 1))
dir=$(echo "${retline}" | cut -d ' ' -f 2)
cd ${dir} || (echo >&2 "Unable to locate output directory: ${dir}";exit 2)
mv debian DEBIAN
cat DEBIAN/conffiles | sed 's/^_prefix//' > DEBIAN/conffilet || (echo >&2 "Unable to update
conffiles in DEBIAN subdir";exit 3) && mv DEBIAN/conffilet DEBIAN/conffiles
mv _prefix/* .
rmdir _prefix/
cat DEBIAN/control | awk -v VERSION="${ver}.${rel}" '{ if ( NR < 8 && $0 == "" ) print
"Version:", VERSION; else if ( substr($0,1,8) == "Depends:" ) print "Depends:";
        else print $0 }' > DEBIAN/controlt || (echo >&2 "Unable to update control in DEBIAN
subdir";exit 3) && mv DEBIAN/controlt DEBIAN/control
cd ..
dpkg -b ${dir} $(echo "${pkg}" | sed 's/^\(.*\)\..*$/\1.deb/')
```

**Figure 112: RPM Conversion Script**

## 17.6  Verifying the Hercules Installation

To verify the Hercules binaries installation following either the Software Management or RPM Installation methods, open a telnet window and run the following command to show the Hercules version:

```
hercules 2>&1 | grep Version
```

The command, like all Linux commands, is case sensitive. It starts the Hercules program and combines the two resulting output streams together ("2>&1") before piping that through a filter ("grep") searching for the text string "Version".



**Figure 113: Verify Hercules Version**

Next display the man pages with the following command:

```
man Hercules
```

This displays the "man" page that is installed as part of the binary package. Use "q" to exit this display. The next figure shows an extract of the Hercules man pages.

If both of these commands return the expected output, then the installation was successful.



**Figure 114: Verify Hercules "man" pages**

## 17.7 Completing the Hercules installation

Once the Hercules binaries have been installed and verified, the following additional tasks should be completed before starting to configure your new Hercules environment.

### 17.7.1 Configuring the Network Interface

In order to support IP routing to a Hercules guest system, a network tunnel is created using the native Linux "tun<n>" device. This device simulates a network layer 3 device and routes IP packets between the Linux operating system and Hercules.

The tunneling device is not active (as a configurable device object) until it is opened by Hercules. The definition of a 3988 CTC device in the Hercules configuration file will cause Hercules to call a special program specifically to start and configure the "tun<n>" device.

This program is called "hercifc" and is installed as part of the Hercules binaries. As we are not running Hercules with root authority and this network device program will require to issue system calls, we need to run some additional commands to allow it to work correctly.

| Command | Description |
|---|---|
| `whereis hercifc` | Display the location of the hercifc program. This "whereis" command is a Linux program that searches in well-known locations for programs or source files. |
| `cd /usr/bin` | Change to the directory containing the "hercifc" program |
| `ls -l hercifc` | List the current attributes of the hercifc program. |
| `sudo chown root:hercgrp hercifc` | Use "sudo" to change the ownership of the "hercifc" program to "root" with a group of "hercgrp". |
| `sudo chmod 4750 hercifc` | Use "sudo" to change the attributes of the "hercifc" program to be setuid to the owner (root) and executable only by root and the group (hercgrp). |

**Table 21: Configuring the Network Interface**



**Figure 115: Configuring the Network Interface**

The result of these commands is to allow the "hercifc" program to execute with root authority (that is what the chmod command above does) but restrict the execution of the program only to the file owner (user "root") and any user connected to the "hercgrp" group.

This will allow the tunnelling device to open and configure correctly when initiated by user "hercules" but reduce the security exposure of having a program setuid-to-root to the minimum.

## 17.7.2  Installing a 3270 Terminal Client

If you don't have your own 3270 Terminal Client, use the procedure detailed in "*Software Management*" on page 147 to install the x3270 Terminal Client included in almost all Linux distributions.
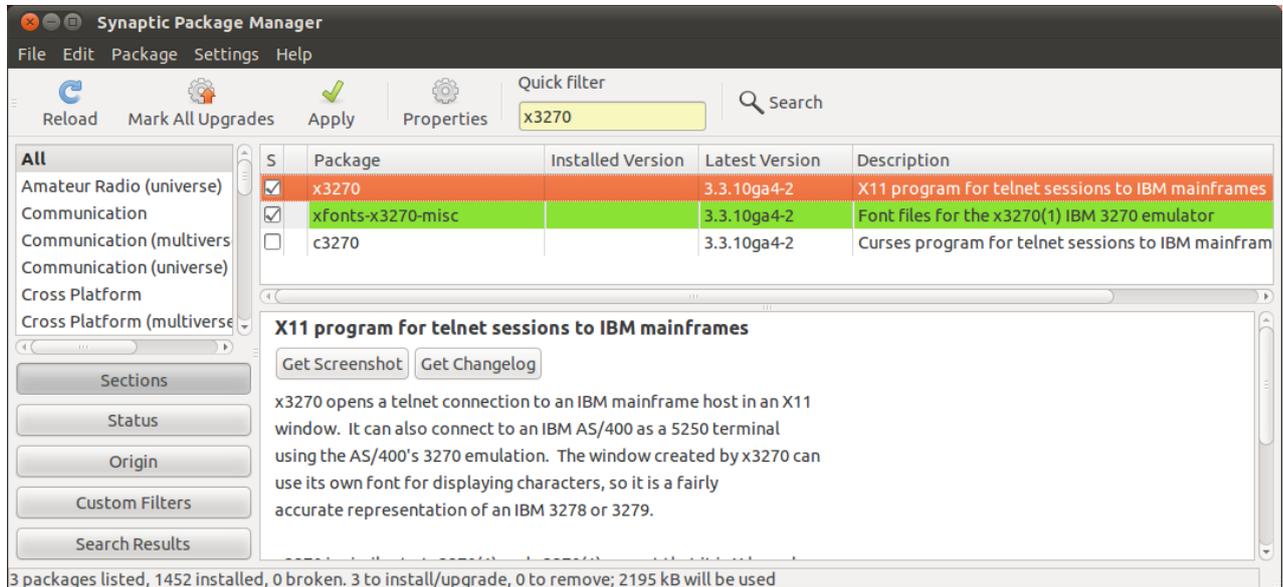


**Figure 116: Installing a 3270 Terminal Client**

# 18. Customization Steps

After the actual installation is complete there are some additional customization steps, some of which are required and some optional, these include:

- Creating the Hercules Configuration File
- Creating the Hercules Startup Script File
- Creating the Hercules Run-Command File
- Creating a Hercules desktop Launcher
- Using the "screen" command to virtualize the Hercules console

These manual customization steps are explained in detail over the following sections.

## 18.1 Directory Locations

Before we create the runtime files to execute our Hercules emulator, it is probably useful to identify what files we are going to create and where they should be created.

Up to this point in the process, we have created the Hercules binaries in locations either controlled by the software package management facility or by specifying a target directory.

Now we are going to create some script and configuration files and we need to identify where we need to create them. Additionally, we will need to identify a location for our target operating system, for example MVS 3.8.

```
\usr
    \bin            Hercules binaries are stored here after a Package Manager installation.
    \lib
        \hercules   Hercules shared object (or linked library) files.
    \local          System defined location for local modifications.
        \bin        Location for local scripts (e.g. start-up script).
\hercules           Completely separate directory for the Hercules guest operating system.
    \mvs            Location for Hercules configuration file and run-command script file.
        \dasd       Location for operating system disk files.
        \tape       Location for operating system tape files.
        \printer    Location for operating system printer files.
        \log        Location for Hercules log files.
```

**Figure 117: Linux Directory Structure Example**

## 18.2  The Hercules Configuration File

When starting the Hercules Emulator from either a telnet shell or via a launcher, you must specify the name of a configuration file as a parameter:

```
hercules –f filename ...
```

where *filename* is the name of the configuration file. The default filename if none is specified during the start-up is 'hercules.cnf'. The name of the default configuration file may be overridden via the environment variable HERCULES_CNF.

The configuration file is an ASCII text file that is used to describe the processor definition, the device layout and any runtime parameters. Details of the format and acceptable directives that can be made within the file are found in the Hercules "User Reference Guide".

Using the directory structure example outlined in "Directory Locations" on page167, we would create this file in:

```
/hercules/mvs/
```

## 18.3  The Hercules Start-up Script File

Although the Hercules Emulator can be started manually from a shell, it is often easier to establish a script file to do this. The script file can then be executed by running it from the shell or creating a launcher to start it from the desktop. We shall use the name "hercules.sh" for our start-up script.

Using the directory structure example outlined in "Directory Locations" on page 167, we would create this file in:

```
/usr/local/bin/
```

### 18.3.1  Create the Hercules Start-up Script File

The following figure shows an example of a Hercules start-up script file using the native Hercules console. You should use one of the installed editors (or "vi" if you are feeling adventurous):

```
#!/bin/sh
#export PATH=/usr/local/hercules/bin:${PATH}
#export LIBPATH=/usr/local/hercules/lib:${LIBPATH}
cd /hercules/mvs
hercules –f mvs.cnf
```

**Figure 118: Hercules Startup Script File**

This example script file contains only two commands:

- The first two commands have been commented out but can be activated if the binaries have been installed into a non-standard location.

- The next command changes the current directory to the location where the configuration and run-command files can be found (see "Directory Locations" on page 167)

- The last command runs Hercules with a specified configuration file.

To create this file using "vi", try the following in a telnet shell:

| Command, Input | Description |
|---|---|
| `cd /usr/local/bin` | Change to the correct directory. |
| `sudo vi hercules.sh` | Use "vi" to edit the new file. Use "sudo" because we probably don't have the authority otherwise. |
| `i` | Enter „insert" mode. |
| `#!/bin/sh` | Type line and <enter>. |
| `#export PATH=/usr/local/hercules/bin:${PATH}` | Type line and <enter>. |
| `#export LIBPATH=/usr/local/hercules/lib:${LIBPATH}` | Type line and <enter>. |
| `cd /hercules/mvs` | Type line and <enter>. |
| `hercules –f mvs.cnf` | Type line and <enter>. |
| `<esc>:wq` | Hit the escape key (which will put you in command mode) and type :wq to write the file and quit. |

**Table 22: Create Hercules Start-up Script File**

## 18.3.2 Making the Hercules Startup Script executable

Now we have created our start-up script file, we need to make it executable. This is necessary as Linux can executable a file only when it is marked as executable. This is a security feature to try to reduce the chance of creating a dangerous command file by mistake.

If you use a standard umask, the file will have been created with attributes of "644", which means that you cannot execute it.

Don't worry too much about what this is if you don't know. You can type "umask" into a shell to see what the current setting is. Using our recommended security environment, we are going to issue two commands:

First we change the ownership of the file to the Hercules userid and its group:

```
sudo chown hercules:hercgrp hercules.sh
```

Now we change the attributes to make it executable:

```
sudo chmod u+x,g+x hercules.sh
```

You can also set the attributes directly using the command:

```
chmod 754 hercules.sh
```

## 18.4  The Hercules Run-Commands File

Hercules also provides the ability to automatically execute Hercules panel commands after start-up via the 'run-commands' file. If the run-commands file exists when Hercules starts, each line contained in the file is read and interpreted as panel command.  We shall use the name "hercules.rc" for our run-commands script.

Using the directory structure example outlined in "Directory Locations" on page 167, we would create this file in:

```
/hercules/mvs/
```

### 18.4.1  Create the Hercules Run-Commands File

In the following example file, an x3270 command is run to start a session for the console and then all the currently valid Hercules panel commands are displayed and MAXRATES is reset. Finally a couple of further x3270 sessions for consoles and TSO terminals are started.

```
sh x3270 –model 3278-3 –once 127.0.0.1:3270 &
?
maxrates
pause 1
sh x3270 –model 3279-4 –once –color 127.0.0.1:3270 &
pause 1
sh x3270 –model 3279-4 –once –color 127.0.0.1:3270 &
```

**Figure 119: Hercules Run-Commands File**

Creative use of the run-commands file can completely automate Hercules start-up and initiate the IPL of the mainframe operating system. For details about the usage of the run-commands file see the Hercules "User Reference Guide".

# 19.  Optional Customization

There are a few additional (optional) customization steps that can simplify the daily work with the Hercules Emulator. These steps are:

- Creating a Hercules Desktop Launcher
- Using the Screen Command

## 19.1  Creating a Hercules Desktop Launcher

Although the Hercules Emulator can be started manually from a telnet shell, it is often easier to create a desktop launcher to do this. The creation of the launcher can vary from Linux variant to variant, but will be similar to the following process for our reference Ubuntu system.

Right-click on the desktop and then click on the "Create Launcher …"
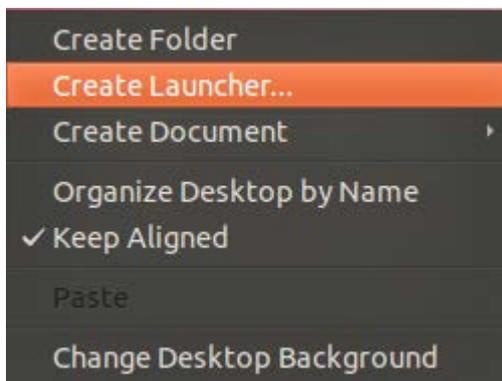


**Figure 120: Create Launcher**

In the following window, enter the required information:

- In the "*Type*" drop-down list, select "Application in Terminal".
- Enter a name for the launcher.
- Enter the Hercules start-up script that you created earlier. In our example, this would be:
  `/usr/local/bin/hercules.sh`
- Enter an optional comment.

You can also select a different icon if you wish.
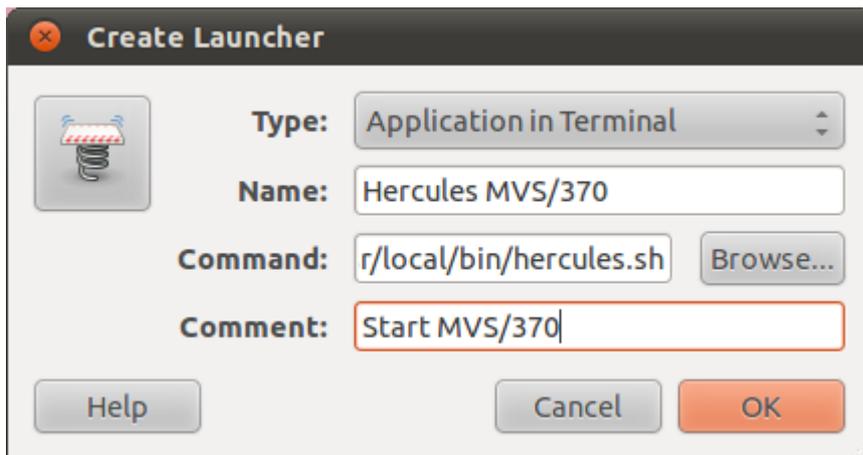
**Figure 121: Create Desktop Launcher Properties**

## 19.2 Running Hercules via Desktop Launcher

Once these command and configuration files have been created, clicking on the launcher will start the Hercules program and the required 3270 sessions:
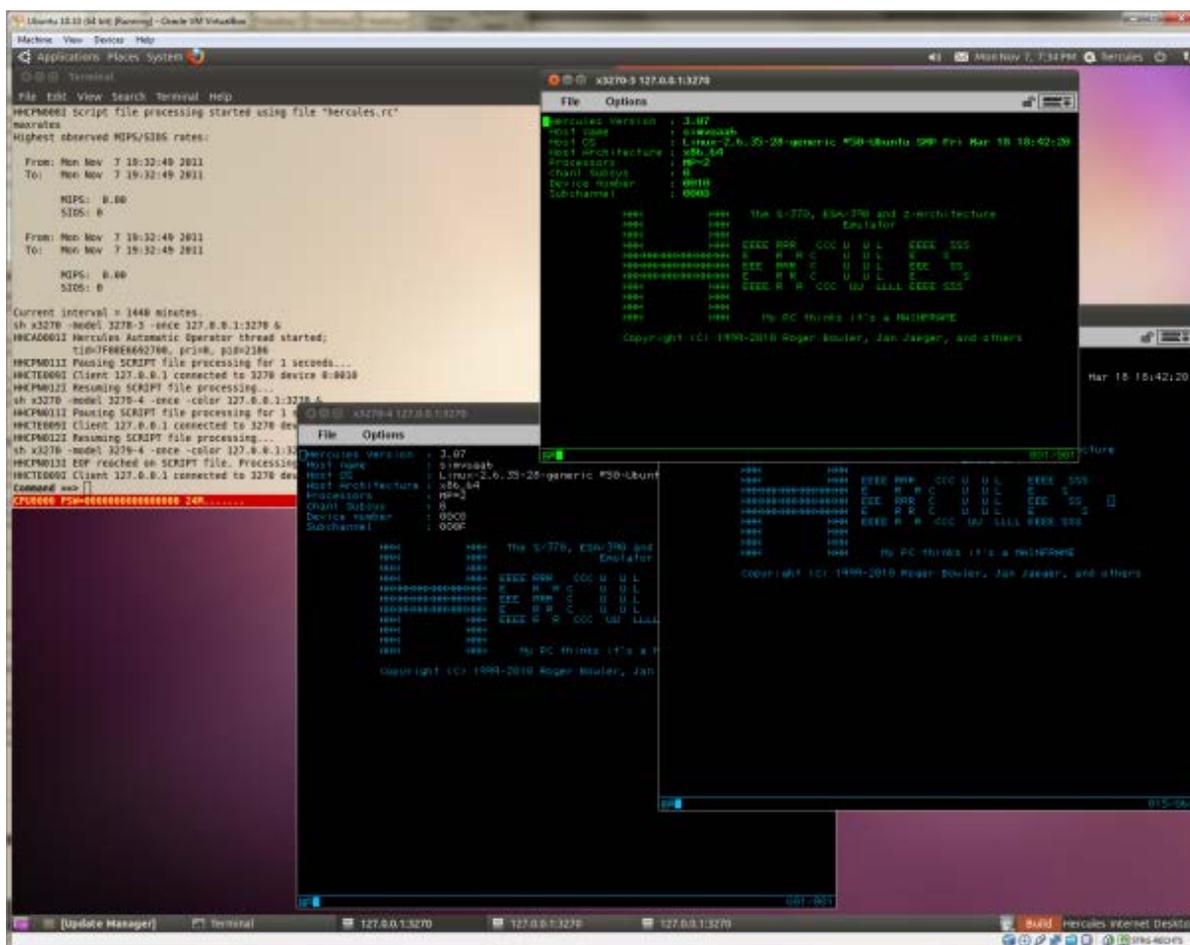


**Figure 122: Running the Desktop Launcher**

## 19.3 Using the Screen Command

A useful extension to the already described methods of running the Hercules command is to augment the chosen method with the Linux "screen" command.

The screen command is a full-screen window manager that builds a virtual terminal providing VT100 emulation and allowing it to be shared between multiple processes.

The advantage of using this facility is that the Hercules console can be created in a "screen" virtual terminal and then moved from one client to another, even to a different host. Of course, only the Hercules console is moved with the virtual screen. The Hercules process itself remains running where it was started.

To use the "screen" command, open a shell window and type the following command:

```
screen –d –r –S Hercules –t "Hercules MVS/370"
```

This command will create a virtual terminal and wait for input to the new window:



**Figure 123: Screen Command**

The first command shown in the figure displays the active virtual screens. Currently there are no screens. Then enter the command shown above to build a new virtual screen with the given name and title.

Once the new virtual screen opens, you can list the available virtual screens which should now include the newly define one active in our shell.
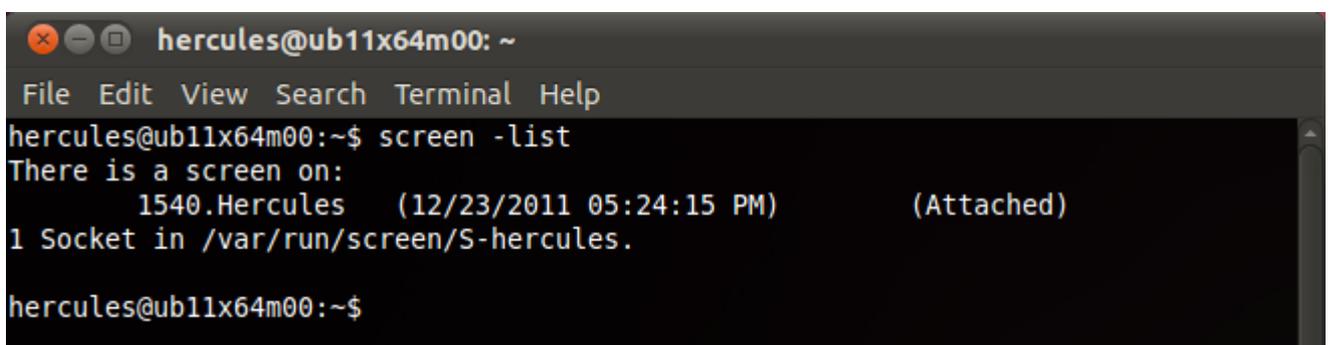


**Figure 124: List Available Screens**

From this virtual screen, you can start the Hercules process. Once started, you can detach this screen using the following commands:

"Alt-a d" from within the virtual screen or with the following command from another process:

```
screen –d Hercules
```

# 20. Installing the Hercules Studio GUI

The Hercules Studio provides a graphical user interface under Linux to the Hercules Emulator itself. Hercules itself has only a semi-graphical user interface, the Hercules Studio gives a full graphical user interface.

The Hercules Studio is the Linux counterpart to the Hercules Windows GUI described in the previous sections. If you are already familiar with the Hercules Windows GUI then you will have no problems working with the Hercules Studio.

Hercules Studio is written and maintained by Jacob Dekel. The Hercules Studio GUI can be downloaded from www.jacobdekel.com/hercstudio.

The Hercules Studio GUI can be treated much like the Hercules binaries themselves. It can be installed using the Software Management package or built from the source tarball that has been made available on the same web site.

There are both 32-bit and 64-bit versions of the binaries package available and they can be installed by clicking on the appropriate link. There is also each version available in RPM format (RedHat, CentOS, Fedora, SUSE, OpenSUSE, …) and DEB format (Debian, Ubuntu, Linux Mint, …).

For our reference Ubuntu 64-bit system, we click on the "*Hercules Studio 1.3.0 in Deb 64-bit format*" link.
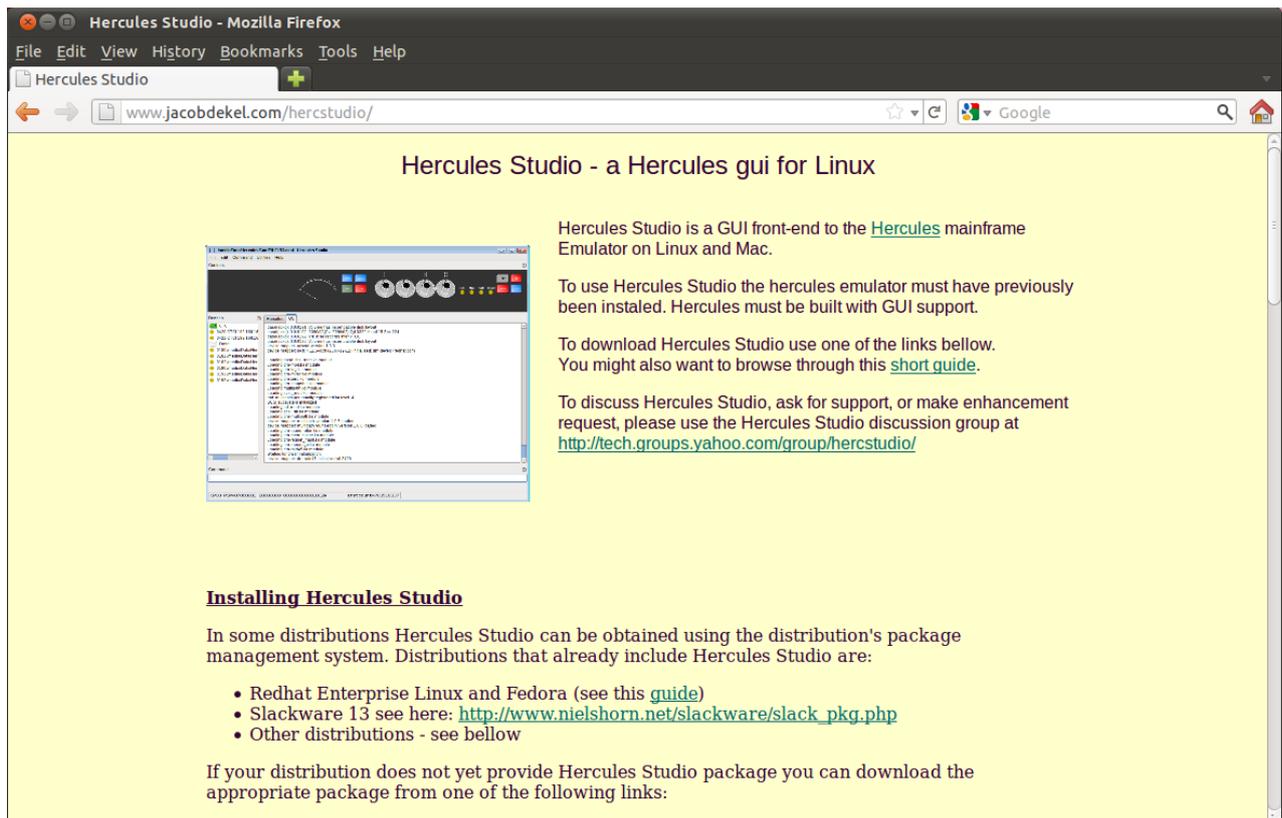


**Figure 125: Hercules Studio WebSite**

## 20.1  Installation Steps

Clicking on the package link should start the Software Management facility which will initiate the installation of the package.
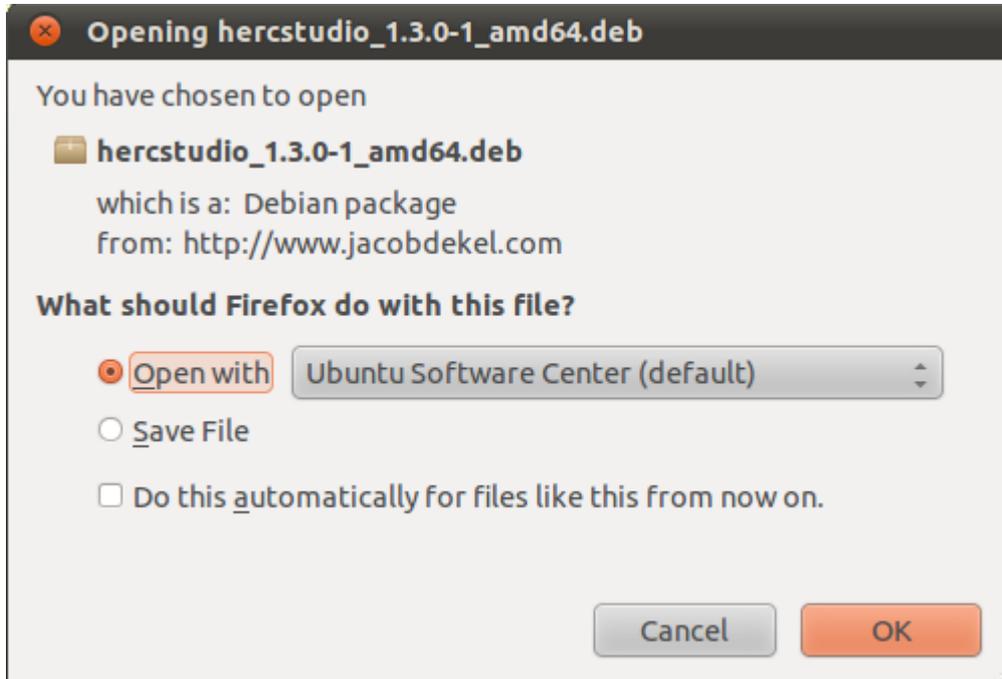


**Figure 126: Initiate Hercules Studio Installation**

Clicking "OK" will start the Software Management installation process. Clicking "Install" will download the selected package and complete the installation:
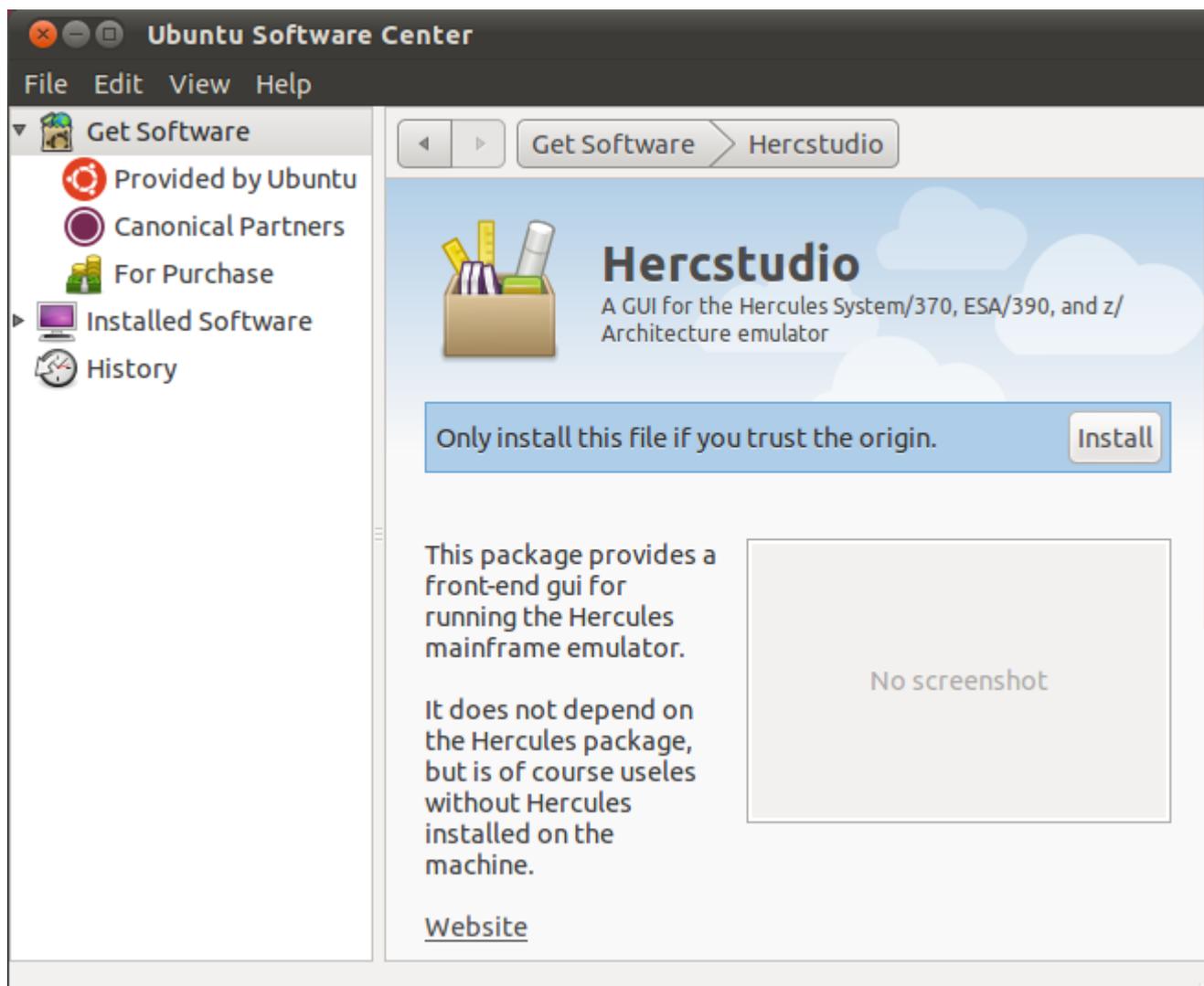
**Figure 127: Hercules Studio Installation**

## 20.2 Running the Hercules Studio GUI

The Hercules Studio GUI can be run from the telnet shell just like the Hercules binaries themselves, but as we have created a start-up script to trigger the Hercules emulator, we can equally trigger the Hercules Studio GUI instead.

To do this, use your favourite text editor to edit the start-up script you created in "Create the Hercules Start-up Script File" and change the line that starts Hercules from:

```
hercules –f mvs.cnf
```

to the following:

```
HerculesStudio -f mvs.cnf
```

The next figure shows the running Hercules Studio. A detailed description about working with the Hercules Studio can be found in the "Operations and Utilities Guide".
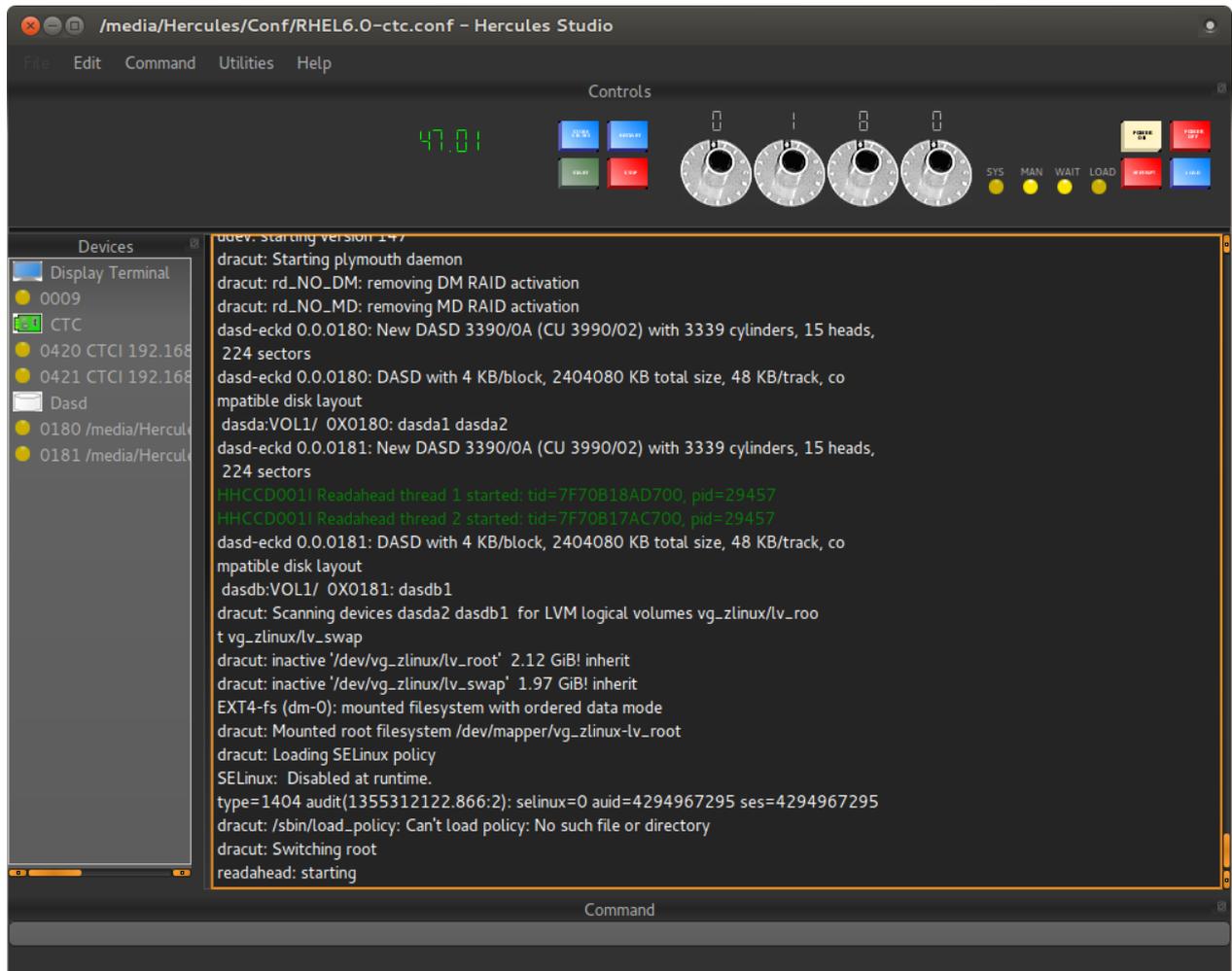


**Figure 128: Running the Hercules Studio**

# Part IV: Mac OS X Installation

# 21. Software Prerequisites

## 21.1 Operating System

Hercules is an open source software implementation of the System/370, ESA/390 and z/Architecture hardware. Hercules itself is not an operating system nor does it emulate a mainframe operating system. The Hercules Emulator runs under Linux on several hardware platforms including the Intel Pentium PC, under various flavours of Microsoft Windows and under Apple Mac OS X. From the point of view of the underlying operating system the Hercules Emulator is just an application program.

Part IV of this guide focuses solely on the installation of Hercules under Apple Mac OS X. Details of other host operating systems are covered in Part II (Windows) and Part III (Linux) in this book. The installation of any hosted operating system and software utilities is beyond the scope of this book.

### 21.1.1 Mac OS X Versions

The Hercules Emulator runs under Mac OS X since Mac OS X 10.2 (Jaguar), native support became available with Hercules Release 2.17. With Mac OS X 10.3 (Panther) a Tun/Tap driver became available. Starting with Mac OS X 10.4 (Tiger) and Hercules Release 3.03, Mac OS X became a suitable platform for the Hercules Emulator too.

The Hercules Emulator supports PPC and Intel Macs with 32-Bit and 64-Bit CPUs. Be aware that there is a performance penalty when Hercules emulates a 64-bit architecture on a 32-bit processor. If you are serious about running 64-bit guests you should probably use Hercules for a 64-bit processor.

Generally it is a good idea to have a current operating system running, maintained with the latest updates. Have also a look at the Tun/Tap driver that is not part of the Mac OS X.

### 21.1.2 Stability

Hercules appears to Mac OS X as just another application program. Although it is possible to run other applications simultaneously with Hercules, this is not recommended.

While the Hercules Emulator is extremely stable, it is impossible to guarantee that other applications will not degrade the Hercules performance. The less software you have running on your Mac, the more stable your emulated mainframe will be running. Prevent any heavy workload changes. Hercules will get no problems indeed, but the guest system will however.

### 21.1.3 System Slow Down

In typical use cases Hercules has a little to no impact on the Mac OS X system overall performance. The impact on performance will mainly depend on the amount of applications and guest systems running at the same time, and how much RAM is available on your Mac OS X system.

### 21.1.4 Installed Software on a Hercules System

It is recommended to have only a minimum of software installed on a Hercules host machine. Installation disk images are available for Mac OS X 10.4 (Tiger), 10.5 (Leopard) and 10.6 (Snow Leopard).

Additional to the software for the Hercules Emulator presented in this guide, it is recommended to have the following software installed:

- Mac OS X 10.4 or later maintained with the latest updates.

- Tun/Tap driver (this is not a part of Mac OS X).

- TN3270 terminal emulation (a Telnet terminal emulation is available with Mac OS X).

- Hercules Studio GUI (optional).

In general though the less software is installed and running, the better the emulated mainframe will run.

## 21.2 Runtime Environments

This chapter describes special runtime environments that act as a base for running the Hercules Emulator. Under Mac OS X the BSD subsystem package is used.

### 21.2.1 BSD Environment

Before using Mac OS X as a Hercules system, you should make sure that the BSD subsystem is installed. On OS X 10.4 (Tiger), this option is installed by default, but on previous versions it was not. You can check for this by looking for the BSD package receipt (BSD.pkg, in "/Library/Receipts"). If this receipt is not present on your system, you must reinstall Mac OS X. When you do this, you must customise the installation by checking the BSD Subsystem check-box.

If you have already installed software updates, you cannot simply go back and reinstall over the existing system because of the potential for version conflicts with the software updates. If the BSD subsystem package is not installed, you will have to do an "archive and install" installation and then reinstall the software updates.

The Mac OS X Developer Tools package (XCode Tools Package) provides additional tools that you will need to install to complete your development environment. These are not part of the default installation. They contain some of the most important tools, such as the compiler (gcc) and debugger (gdb) in the Unix development component package. This package is provided for compatibility with shell scripts and makefiles that require access to the Developer Tools. Starting with Mac OS X 10.7 (Lion) the Command Line Tools package enables Unix style development. The Developer Tools are essential for you only to build Hercules by yourself.

The terminal application is located in the Utilities folder (which is within the Applications folder at the root level of your hard drive).

Once you have a terminal window open, you can take advantage of a basic selection of common tools and lots of Hercules line command tools, assuming that Hercules was installed.

With the BSD subsystem package installed, a look through "/bin" and "/usr/bin" should reveal a familiar environment for BSD and Unix user. The Hercules command line tools are installed through "/usr/local/bin".

### 21.2.2 Runtime Security

The Hercules Emulator can run under a root user (any user with a UID of 0), but it is recommended to run it under its own user. To do this you have to create a new user under which to run the Hercules Emulator.

### 21.2.3 Firewall

The Hercules Emulator acts as server application, you must allow incoming connections in the Mac OS X firewall for the Hercules Emulator. The best time to do this is the first time you start Hercules. The Mac OS X firewall requests you to authorise the access for the Hercules Emulator.

**Note:** On Mac OS X 10.4 (Tiger), open the TCP ports 3270, 3505, 3990 and 8081 (depending on your ports set in the Hercules configuration file) in the firewall of your Mac. On Mac OS X 10.5 (Leopard) and above, you'll just have to click on "Always Allow" the first time you start the Hercules Emulator if the firewall is set.

By default the Hercules Emulator use TCP Port 3270 for terminal access and 8081 for the internal HTTP server.

### 21.2.4 Time Machine

If you use Time Machine, exclude the Hercules guest DASDs from the backup. Any change to a guest DASD file results in the entire file being backed up. When you have a guest with 20 GB DASD files space, having Time Machine back it up every hour, you can fill a hard disk very quickly.

Exclude all guest DASD files from Time Machine to prevent problems with your guest and to save space on your backup media, you should not back up guest DASD files with Time Machine when the guest system is running. But in general backup your data, backups are one area where paranoia is prudent!

There are many ways to do a backup, depending on how much data you have, how often it changes and how valuable it is for you.

### 21.2.5 Endianness

If you move a Hercules guest system from a PPC to an Intel Mac, the CCKD DASD files are converted once at the Hercules start-up:

```
HHCDA020I dasd/mvsres.148 cyls=560 heads=30 tracks=16800 trklen=19456
HHCCU101I 0148 file[0]: converting to little-endian
```

If you move the CCKD DASD files back to a PPC Mac again then you will see the following messages:

```
HHCDA020I dasd/mvsres.148 cyls=560 heads=30 tracks=16800 trklen=19456
HHCCU101I 0148 file[0]: converting to big-endian
```

This is normal behavior all the time you switch between processor architectures that use the little-endian format (the Intel format), like Intel x86 and x86-64, and processor architectures that use the big-endian format (the Motorola format or network order format), like Motorola PPC and 680x0 or IBM PPC and POWER.

The mainframe System/360 and its successors such as System/370, ESA/390 and z/Architecture hardware use the big-endian format too.

**Note:** See Hercules utility "CCKDSWAP" for more information.

## 21.3 Hercules Emulator

The Hercules Emulator consists of the following mandatory and optional components.

- Hercules Emulator package.

- Tun/Tap driver package.

- 3270 client application.

- Hercules Studio GUI (optional).

## 21.3.1 Hercules

The Hercules executables are the heart of the emulator and a mandatory component. This is the software implementation of the System/370, ESA/390 and z/Architecture mainframe hardware and processor machine code instruction set.

Hercules runs as a shell program and comes with a semi-graphical display in a shell terminal (the Hercules Hardware Console - HMC) consisting of two screens, switched between using the Esc key. When the Hercules HTTP server is running, Hercules can also be operated via a web browser.

The following figure shows the initial display in the Hercules Console Window.



```
● ● ●                    Terminal — hercules — 80×24
HHCDA020I dasd/work01.170 cyls=959 heads=12 tracks=11508 trklen=35840       +
HHCDA020I dasd/work02.180 cyls=885 heads=15 tracks=13275 trklen=47616
HHCDA020I dasd/work03.190 cyls=1113 heads=15 tracks=16695 trklen=56832
HHCDA020I dasd/mvscat.191 cyls=1113 heads=15 tracks=16695 trklen=56832
HHCDA020I dasd/pub000.240 cyls=555 heads=30 tracks=16650 trklen=19456
HHCDA020I dasd/mvsdlb.248 cyls=560 heads=30 tracks=16800 trklen=19456
HHCDA020I dasd/pub002.280 cyls=1770 heads=15 tracks=26550 trklen=47616
HHCDA020I dasd/cbt000.340 cyls=555 heads=30 tracks=16650 trklen=19456
HHCDA020I dasd/cbt001.341 cyls=555 heads=30 tracks=16650 trklen=19456
HHCDA020I dasd/cbt002.342 cyls=555 heads=30 tracks=16650 trklen=19456
HHCDA020I dasd/cbtcat.343 cyls=555 heads=30 tracks=16650 trklen=19456
HHCDA020I dasd/src000.348 cyls=555 heads=30 tracks=16650 trklen=19456
HHCDA020I dasd/src001.349 cyls=555 heads=30 tracks=16650 trklen=19456
HHCDA020I dasd/src002.34a cyls=555 heads=30 tracks=16650 trklen=19456
HHCDA020I dasd/srccat.34b cyls=555 heads=30 tracks=16650 trklen=19456
HHCCP002I CPU0000 thread started: tid=102560000, pid=35311, priority=15
HHCCP003I CPU0000 architecture mode S/370
HHCPN001I Control panel thread started: tid=7FFF70557CC0, pid=35311
HHCTT001W Timer thread set priority -20 failed: Permission denied
HHCTT002I Timer thread started: tid=102663000, pid=35311, priority=15
HHCAO001I Hercules Automatic Operator thread started;
          tid=10296C000, pri=16, pid=35311
Command ==>
CPU0000 PSW=0000000000000000 24M.......                        instcount=0
```

**Figure 129: Hercules Console Window**

The next figure shows the Hercules device and status display accessed with the Esc key.
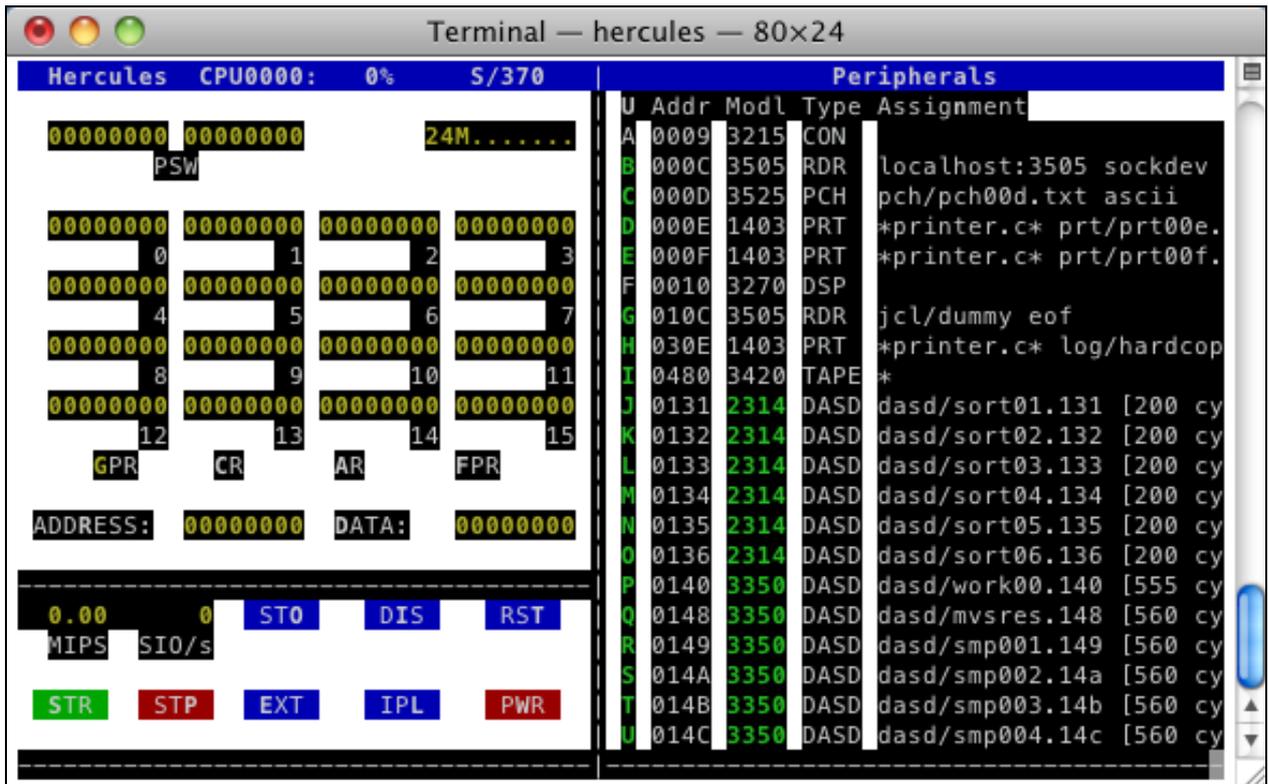
**Figure 130: Hercules Device and Status Display**

The following figure shows the Hercules web browser interface that can be accessed when the Hercules HTTP server is configured and running. This is the only graphical user interface beneath the Hercules Studio GUI for Hercules on Mac OS X today, but it is very useful to get access to the Hercules console.
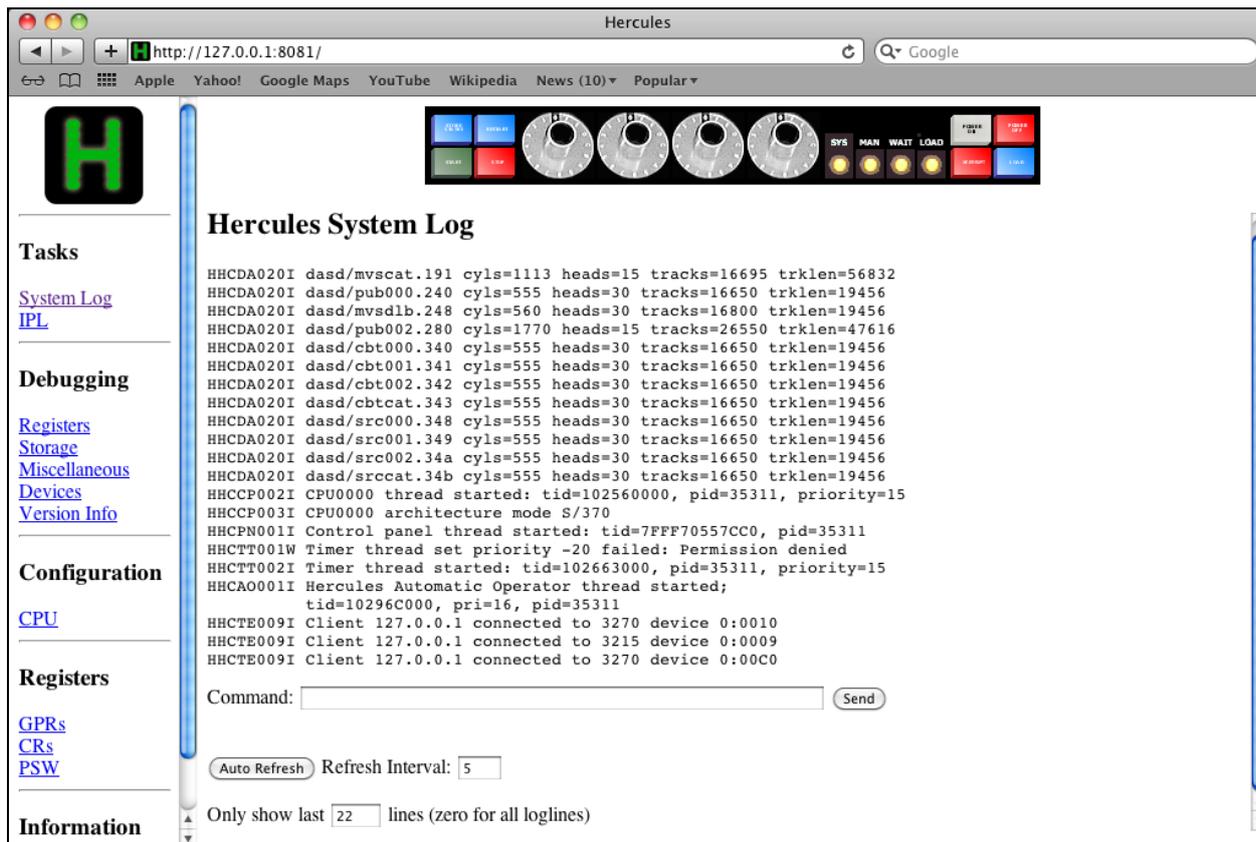
**Figure 131: Hercules Web Browser Interface**

Optionally the Hercules Studio GUI is available. Hercules Studio GUI is not part of the Hercules Emulator and it is necessary to build Hercules yourself to support this GUI.

## 21.3.2 Tun/Tap Driver

You need a tunnel driver to provide networking from your Hercules guest system. If you have not installed it already, then install a Tun/Tap driver. Hercules can use this driver to provide networking functions to the guest system running on the Hercules emulator.

The Tun/Tap driver from Mattias Nissler is shipped with the Hercules Emulation installation disk images. The installation package (http://downloads.sourceforge.net/tuntaposx/tuntap_20111101.tar.gz) should work with Mac OS X 10.6 (Snow Leopard) and above. Older versions of the Tun/Tap Driver and the source code are available also (see http://sourceforge.net/projects/tuntaposx/files/). Have also a look at the web site http://tuntaposx.sourceforge.net.

## 21.4 Additional required and optional software

Additional software to the components described above are also required for practical use of Hercules (e.g. 3270 client) or makes common tasks easier (e.g. netcat, submit).

### 21.4.1  3270 Client

You need a 3270 client to get access to guest systems like MVS 3.8 or VM 6.0 for System/370. For virtual 3270 consoles and 3270 terminals a 3270 client software application is required. The 3270 client can run on the same machine as Hercules or on any other Mac OS X, iPhone, iPad, Linux or Windows system with a TCP/IP connection to the system where Hercules is running.

One of the most common 3270 clients for Mac OS X is the Brown TN3270 terminal emulation. The 3270 client is written and maintained by Peter DiCamillo. Brown TN3270 terminal emulation is a freely available 3270 client and can be obtained from http://www.brown.edu/cis/tn3270/.

Mochasoft TN3270 for Mac OS X is a nice 3270 client for Mac OS X too, but not freely available. Currently available is a free 30-day trial and the single user license costs approximately 30 USD. Mochasoft TN3270 is available for iPhone/iPad or Android, Blackberry and Windows Vista, Windows 7 or Windows 8 and some other platforms too. Mochasoft TN3270 can be downloaded from the following web site: http://www.mochasoft.dk/tn3270macx.htm.

EMTec ZOC Terminal is a 3270 client for Mac OS X and Windows. Currently available is a free 30-day trial and the single user license costs approximately 80 USD. EMTec ZOC Terminal can be downloaded from web site http://www.emtec.com/zoc/index.html.

Suite 3270 is a nice and freely available 3270 client for BSD and Unix, therefore usable for Mac OS X too. Build c3270, s3270 and pr3287 from the available source code and you get a stable 3270 client and a 3287 printer. The 3270 client and 3287 printer is maintained by Paul Mattes. Suite 3270 can be downloaded from web site http://sourceforge.net/projects/x3270/ or installed with the MacPorts or Homebrew package management software.

Because the 3270 client is an independent piece of software there are no version requirements. You can use any stable release of a 3270 client although it is recommended to always run with a current release.

Additional information can be found in the Hercules "General Information" manual, chapter "TN3270 Client".

### 21.4.2  Telnet Client

If you are interested in using Linux as a guest system a telnet client is required to get access to. A telnet client and SSH client is part of Mac OS X. The Brown TN3270 terminal emulation acts as a telnet client too.

### 21.4.3  Graphical User Interface

Hercules Studio GUI is an optional graphical user interface for the Hercules Emulator. Up to version 3.07 is was necessary to build Hercules by yourself to enable GUI support to use the Hercules Studio GUI. Beginning with version 3.08 Hercules has enabled GUI support for all platforms by default. Hercules Studio GUI is written and maintained by Jacob Dekel.

Have a look at web site http://www.jacobdekel.com/hercstudio. Additional information can be found in the Hercules "General Information" manual, chapter "Hercules Studio".

# 22. Installing the Hercules Emulator

## 22.1 Installation Preparation

Before we start to install the Hercules binaries and associated software, a word needs to be said about security.

Although installing Hercules in a single-user Mac OS X system, this does not mean to remove the necessity of defining a security environment for it and it is recommended that the security environment is established in order to better show how Hercules relates to the host operating environment.

To create the security environment for Hercules, we will establish a new standard user account "hercules" that will primarily be used for file ownership. The created user can also be used to run the Hercules process.
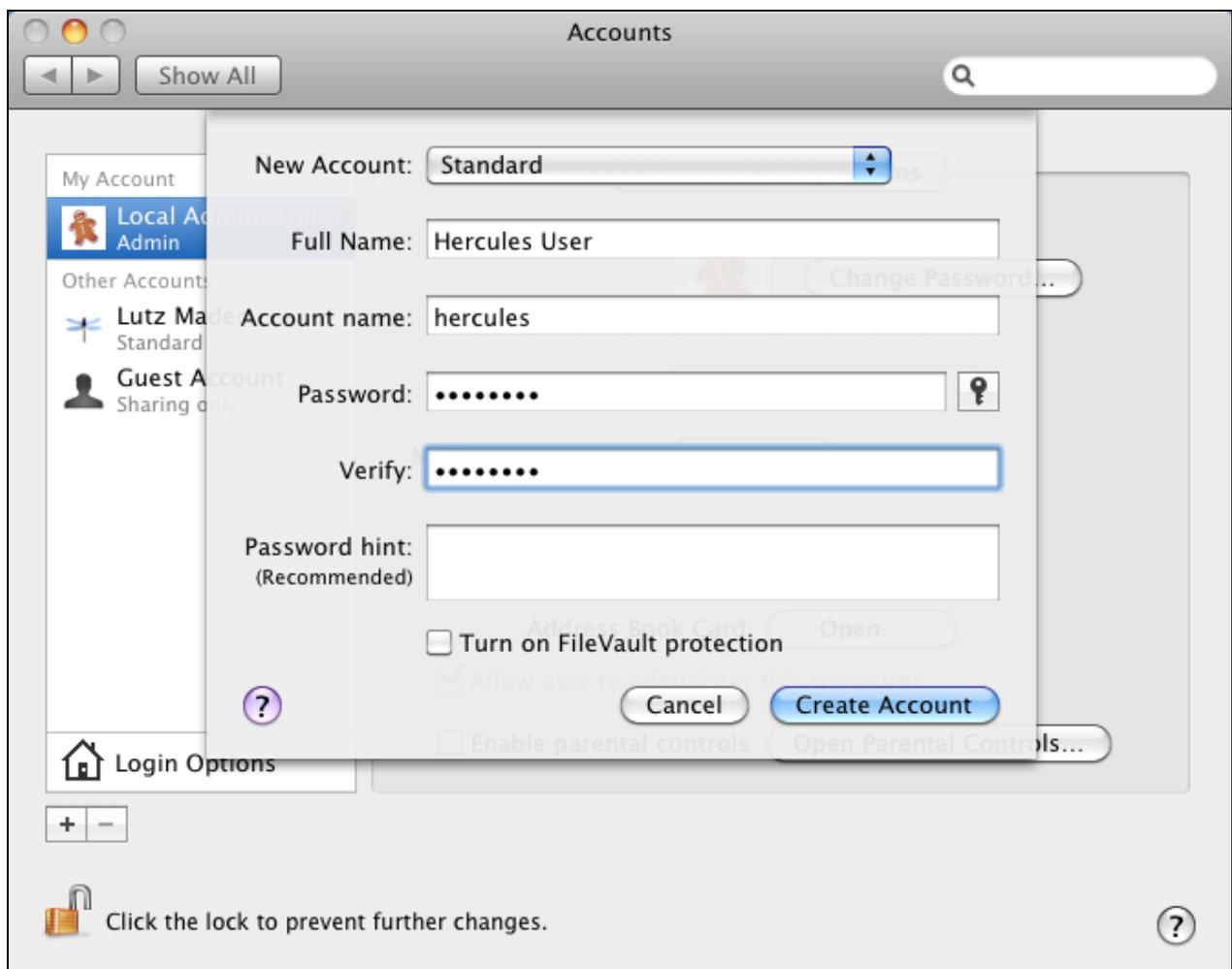


**Figure 132: Add Hercules account**

To reflect to the Hercules package installation file use "dseditgroup" to assign the Hercules user to the "wheel" group as shown in the following command:

```
$ sudo dseditgroup -o edit -a hercules -t user wheel
```



**Figure 133: Assign the wheel group**

Read the chapter "Configuring the Network Interface" to get more information about the Hercules network interface.

## 22.2  Installation methods

There are two ways to install Hercules on Mac OS X and your choice of method may depend on your version of Mac OS X and your attitude to the Hercules software itself.

- Download and install the Hercules package from a disk image.

- Download the Hercules source code and build the binaries by yourself.

If such a plug-and-play approach to software installation does not suit you or you require greater flexibility when installing software packages, the Hercules web site at http://www.hercules-390.eu contains packages for both binary and source code installation.

Both of the above techniques result in the currently released stable version of the Hercules binaries being installed. This is normally the best practice, so either of these two techniques is normally recommended.

Some people however, need the latest software being developed by the Hercules engineers and more up-to-date packages can also be implemented.

## 22.3 Software installation

This is the easiest and the preferred way to install Hercules on Mac OS X.

**Note:** Unfortunately the available ready-to-run binaries are not available at this time. As of writing this chapter the only way to install Hercules on Mac OS X is building Hercules from source as described in chapter 22.4.1 "Building from Source".

### 22.3.1 Downloading the Binaries

The ready-to-run binaries can be downloaded from web site http://www.hercules-390.eu. For Mac OS X users there are currently three installation disk images available.

- Mac OS X 10.4 (Tiger) universal binary version, 32-bit Intel and PowerPC (file hercules-v.rr-tiger.dmg).

- Mac OS X 10.5 (Leopard) universal binary version, 32- and 64-bit Intel and PowerPC (file hercules-v.rr-leopard.dmg).

- Mac OS X 10.6 (Snow Leopard) universal binary version, 32- and 64-bit Intel (file hercules-v.rr-snowleopard.dmg).

"v.rr" in the filename specifies the version and release of the Hercules Emulator.

There is no difference between these installation packages in general, except of the supported Mac OS X. Which one of these installation disk images is the right one for you depends on your Mac OS X only.

The Hercules installation disk image for Mac OS X 10.6 should work with Mac OS X 10.7 (Lion) and Mac OS X 10.8 (Mountain Lion) too. But do not install the Tun/Tap driver contained in the Hercules installation disk image, see chapter 23.1 "Installing the tun/tap driver" for more information.

### 22.3.2 Installation Steps

To start the installation dialog, open the Hercules installation disk image .dmg file and double click the .pkg package file. Hercules itself, without a mainframe operating system installed, uses approximately 12 MB of disk space of your system disk "Macintosh HD". A welcome window is presented first.

**Figure 134: Introduction window**

Click on "Continue" to proceed with the next step.
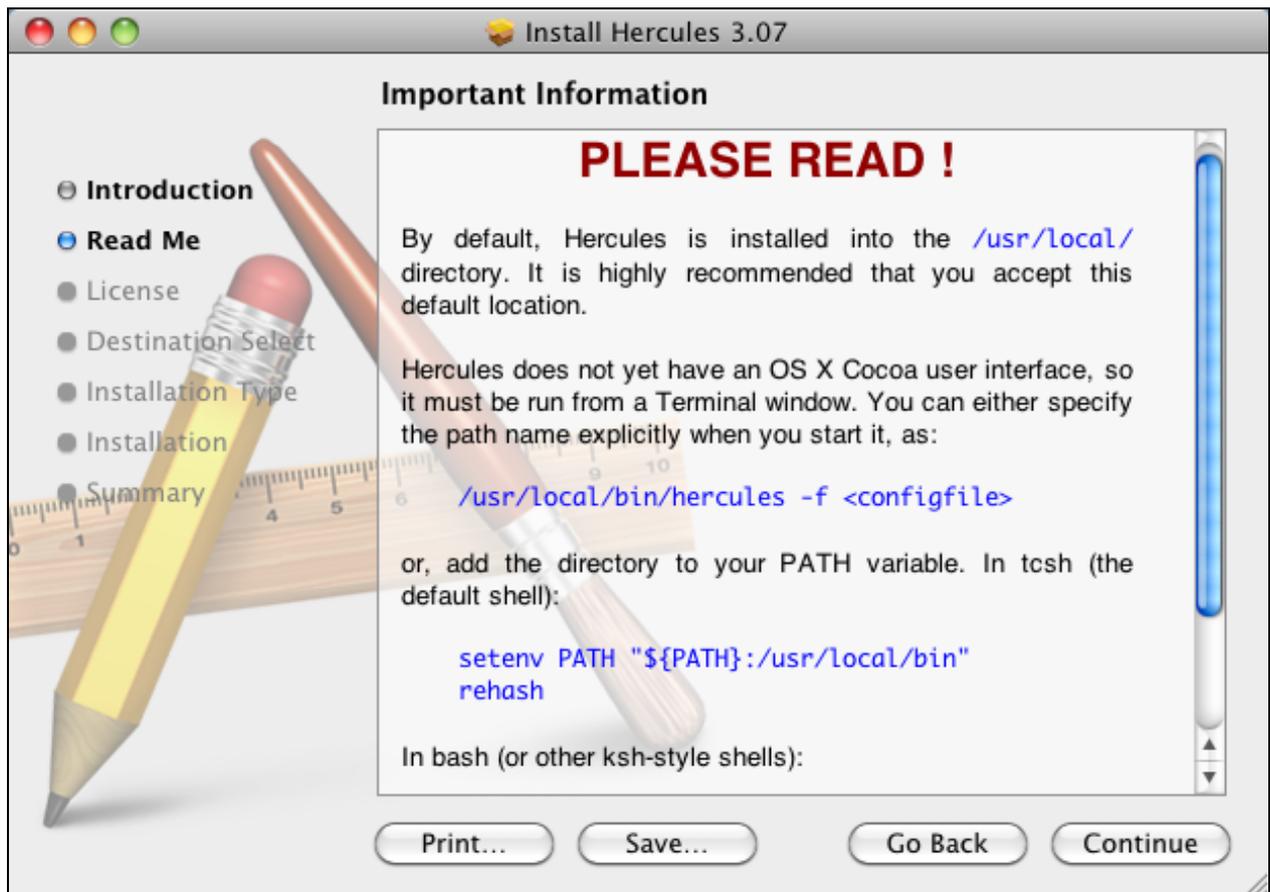
**Figure 135: Read Me Window**

Read the important information and click on "Continue" to continue the installation process. If you are not familiar with Unix or BSD save the information first, click on "Save...".
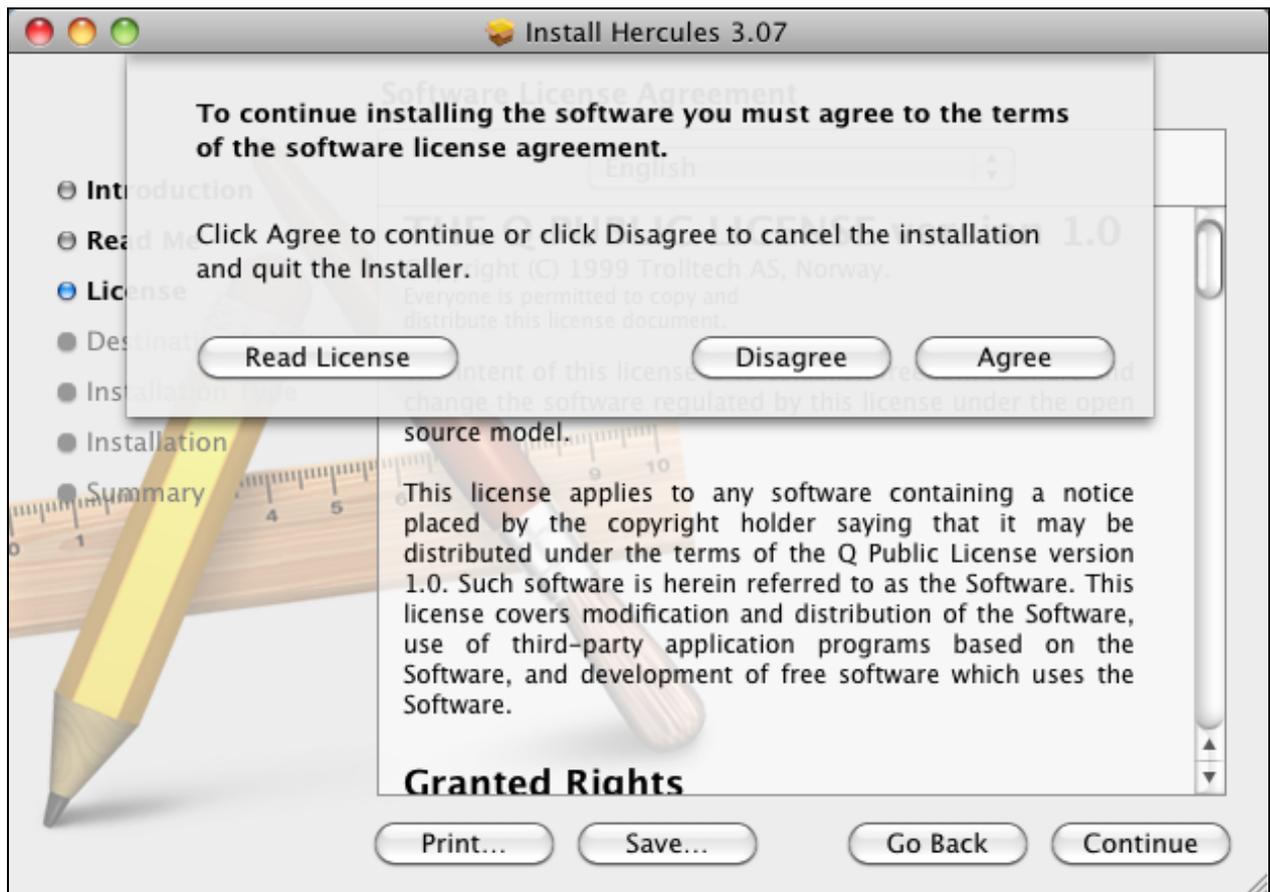
**Figure 136: License Window**

Read the licence agreement and click on "Continue", then click on "Agree" to accept the licence agreement and to continue the installation process.
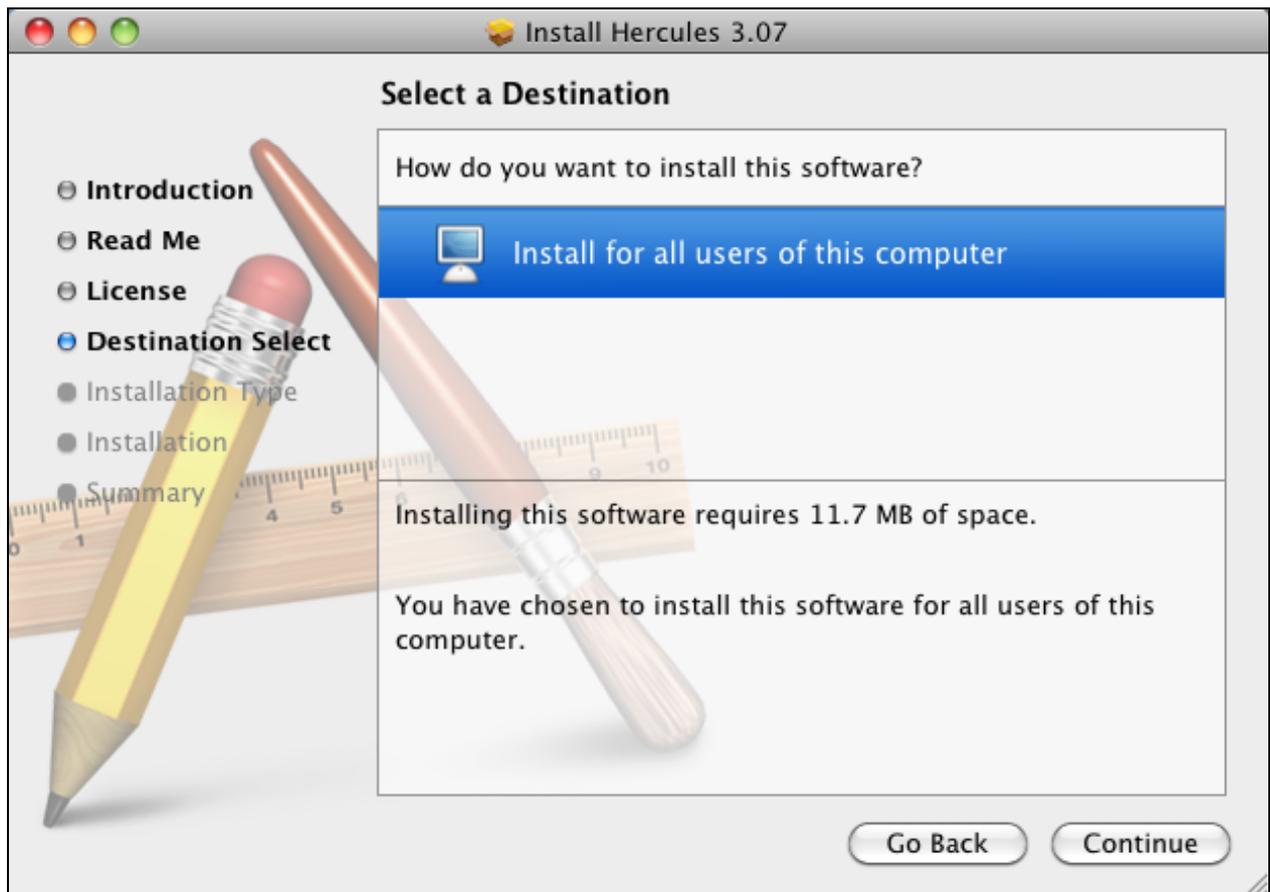
**Figure 137: Destination Selection Window**

Click on "Continue" to continue the installation process.
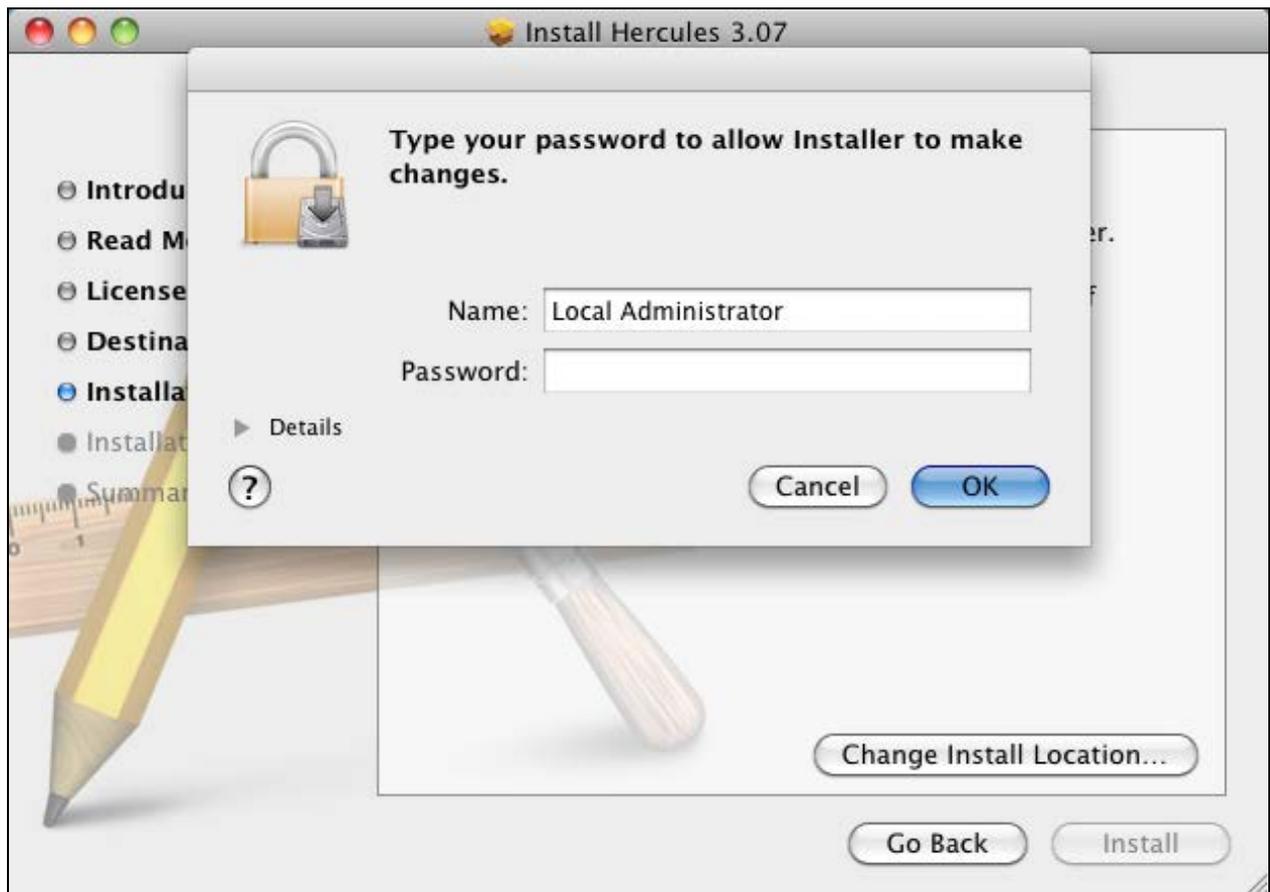
**Figure 138: Installation Type Window**

Click on "Install" to continue the installation process and then enter the name for an administrator and the password and click "OK" to perform the standard installation process. You cannot select the installation destination at this time. The installer now begins to copy files to the system disk "Macintosh HD".
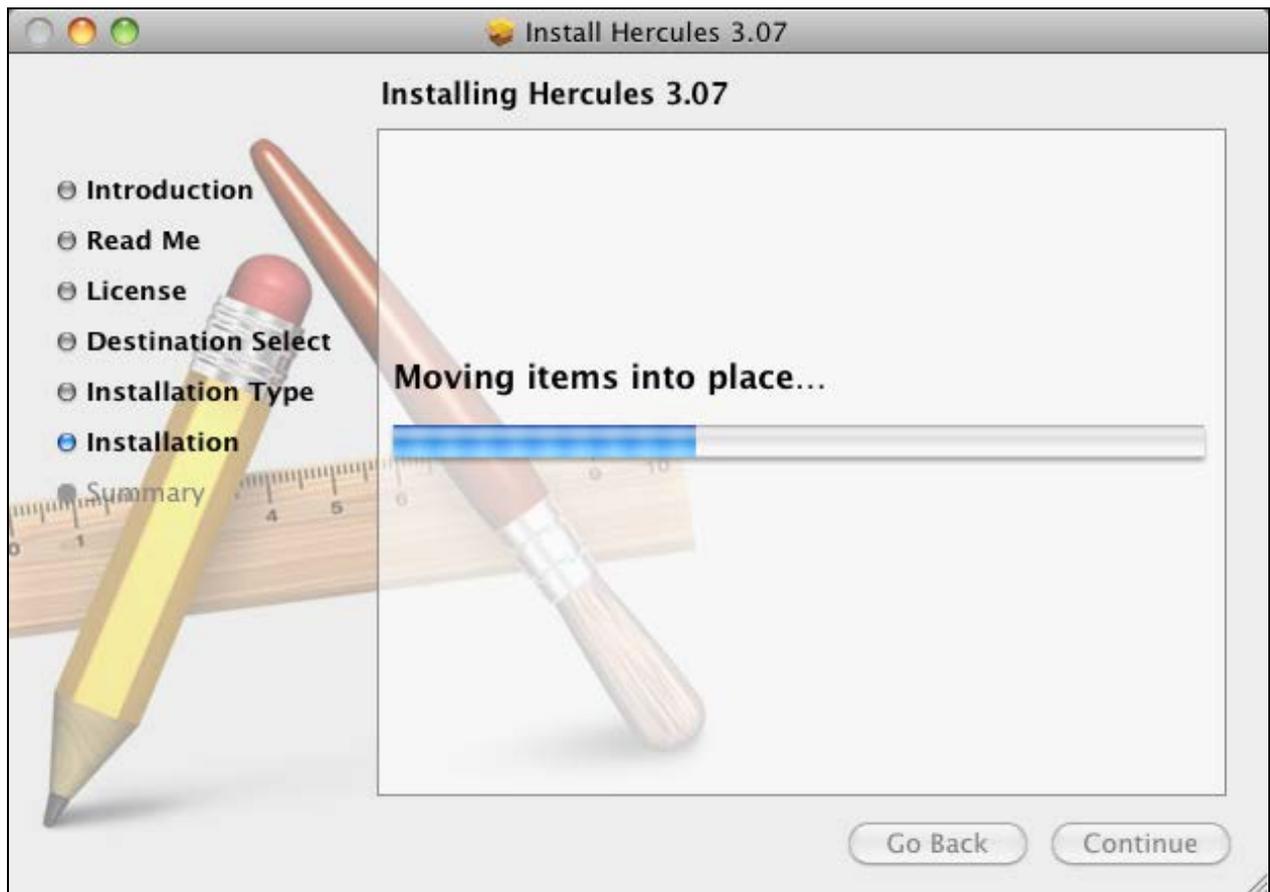
**Figure 139: Installation Window**

After a few seconds the installation process will finish and present the final window.
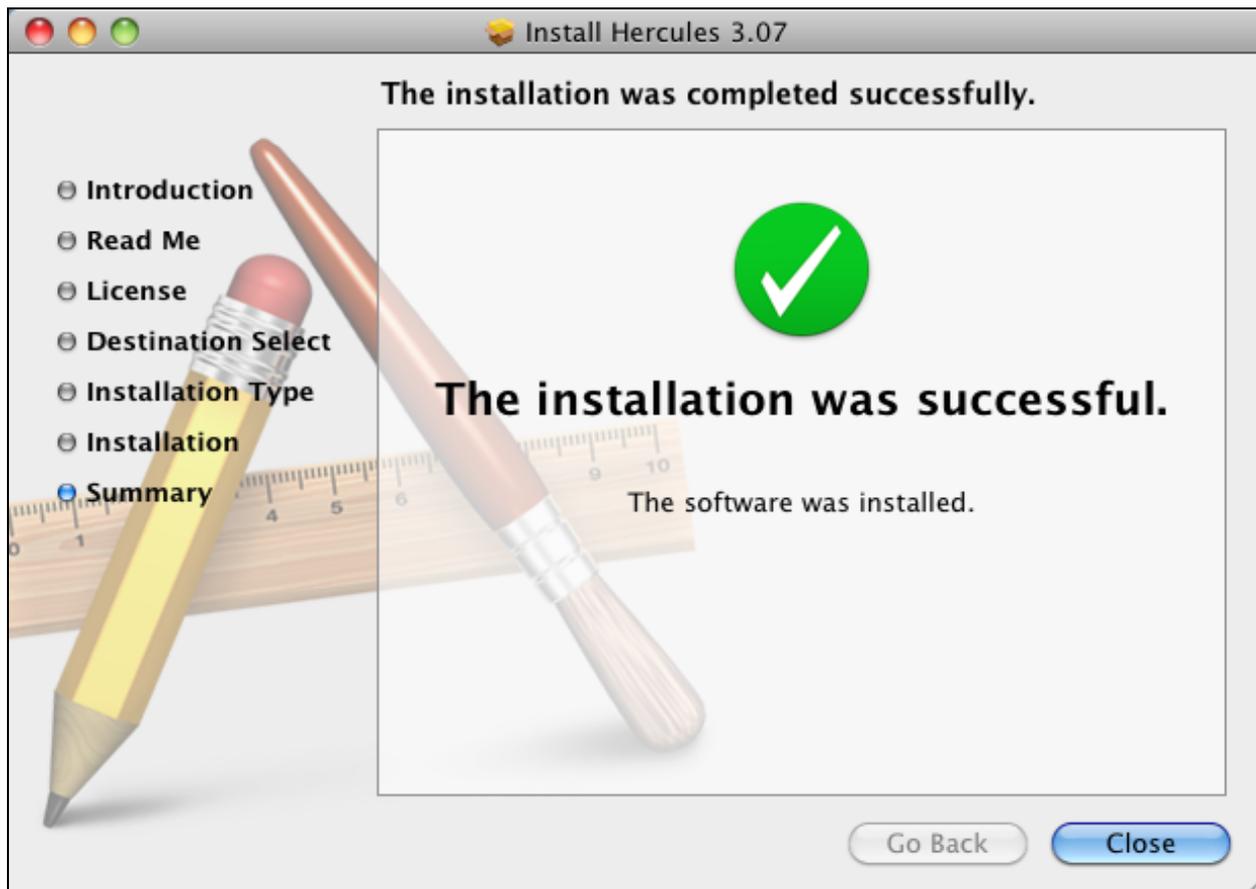
**Figure 140: Summary Window**

Click on "Close" to terminate the installation process and proceed to customise your installation as described in the following sections. You can skip to chapter 24 "Customization steps" or to chapter 25 "Installation Verification Procedure" to continue.

If you need the tunnel driver or link layer driver to provide networking from your Hercules guest system and have not installed it already, a tun/tap driver is in the tuntap folder. Hercules can use this driver to provide networking functions to the guest system running on the emulator. Read chapter 23.1 "Installing the tun/tap driver" to get more information about the installation package in the tuntap folder.

## 22.4  Building from Source

The second installation option is to build the Hercules binaries from the available source distribution. Despite the extra effort involved, this method has a significant advantage over the more automated installation options.

The configure and build process allows the installer to specify a number of configuration and optimization flags to control how the binaries are built.

Configuration options can provide a mechanism to control where the Hercules binaries are installed. This can be used to easily integrate one of the daily snapshots of the Hercules source code available from Dave Wades build site (see next chapter).

Optimization options can build the binaries in such a way that they better exploit the hardware on which they run. Finally, this is the only method I assume a true dyed-in-the-wool system programmer would sanction!

## 22.4.1 Downloading the Source

The source code can be downloaded from the Hercules web site (http://www.hercules-390.eu). The file is named "hercules-v.rr.tar.gz", where the "v.rr" in the filename specifies the version and release of the Hercules Emulator

The developers source code snapshots, that are updated daily whenever the source code changed, are available from Dave Wades website at  http://www.smrcc.org.uk/members/g4ugm/snapshots/. Read the "README.SVN" file included with the source code for additional and updated instructions to build a development version.

To unpack the source code archive file double click the .tar.gz archive file. This will extract all the required files into a new folder in the current folder location that should have the name "hercules-v.rr", the name of the source package file without the suffixes, where "v.rr" specifies the version and release.

Open the new folder and drop "README.OSX" to the TextEdit application to get the updated additional how-to build information.

You are welcome to the undiscovered country, now we enter the environment a Mac OS X user has never seen before, welcome home BSD and Unix user.

**Note:** Before Apple came up with the user-friendly term folder to represent a holding tank for files, folders were called directories. This means the same thing; in this part you'll encounter the term folder exclusively. In any discussion of Unix and BSD, "directory" is simply the correct term.

Open the Terminal application, located in the Utilities folder (which is within the Applications folder at the root level of your system disk "Macintosh HD"). You are inside your Mac OS X now.The Terminal application is the doorway to the BSD.

**Figure 141: Utilities Folder**

Enter "cd " (cd with a trailing blank), switch to your previous download folder, select the new folder and drop the folder to the Terminal. Select the Terminal window and press the Enter key, to change into the folder with the extracted source code.
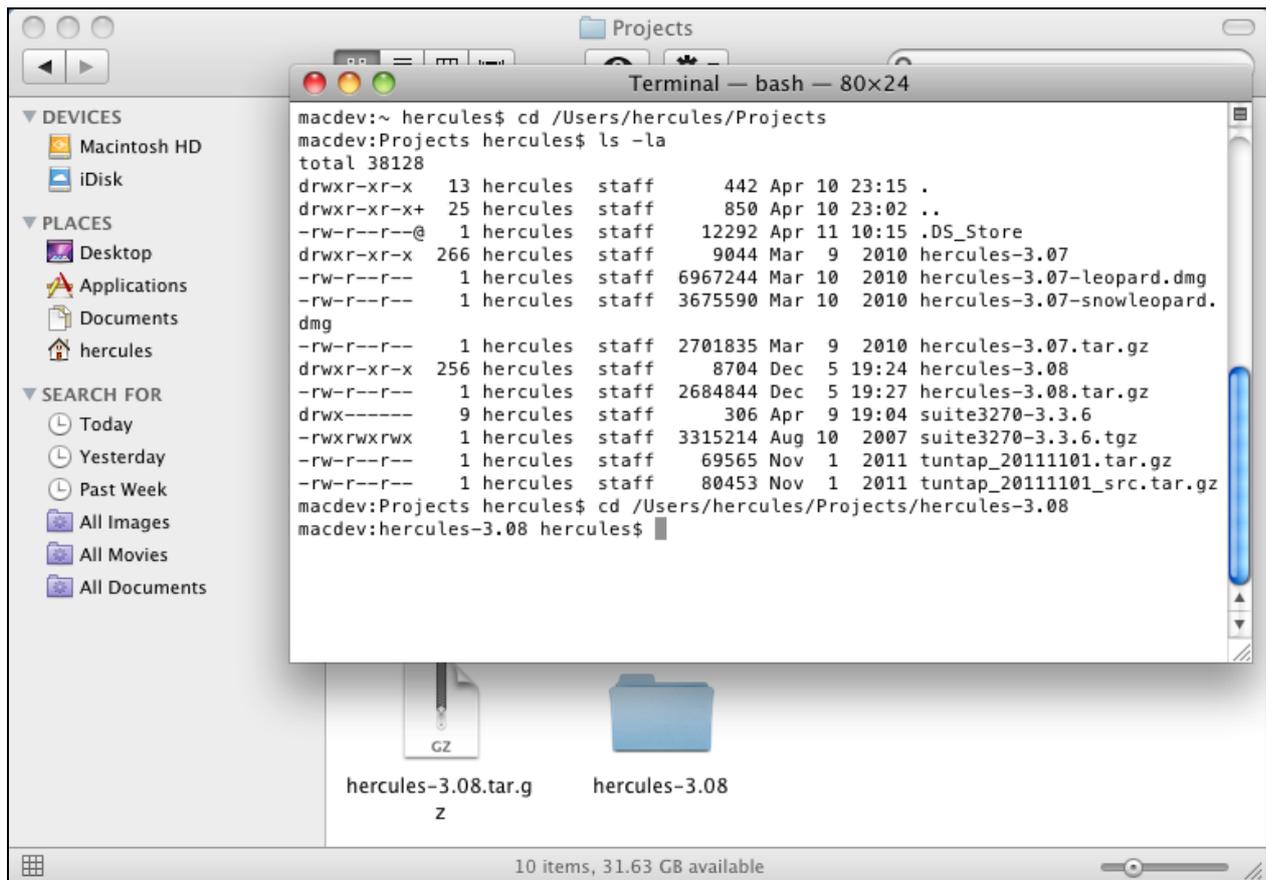
**Figure 142: Project Folder**

## 22.4.2  Verify the Environment

Issue the following command to verify you have all of the correct versions of all of the required packages installed now:

```
$ ./util/bldlvlck
```

Until Hercules 3.07 issue the following command.

```
$ ./util/cvslvlck
```

```
000                    Terminal — bash — 80×24
macdev:~ hercules$ cd /Users/hercules/Projects/hercules-3.08
macdev:hercules-3.08 hercules$ ./util/bldlvlck
This utility will check the level of various utilities needed to build
hercules. Checking is done against versions that are KNOWN to work.
This doesn't mean a build will NOT succeed with older versions
of the utilities, but will give a hint as to what package may need
an upgrade if the build ever fails with some odd reason.


OK      svn requires 0, found 1.6.17

OK      autoconf requires 2.5, found 2.61

OK      automake requires 1.9, found 1.10

OK      flex requires 2.5, found 2.5.35

INSTALL gawk not found
        URL: http://www.gnu.org/directory/gawk.html

OK      gcc requires 2.95, found 4.2.1

INSTALL gettext not found
        URL: http://www.gnu.org/directory/gettext.html
```

**Figure 143: Verify the packages**

```
000                    Terminal — bash — 80×24
OK      gcc requires 2.95, found 4.2.1

INSTALL gettext not found
        URL: http://www.gnu.org/directory/gettext.html

OK      grep requires 0, found 2.5.1

OK      libiconv requires 1.8, found 1.11
        For first-time Libiconv installs, a recompile and reinstall of gettext
        is recommended.  Source: libiconv-1.8/README

Apple/Darwin ==> custom version parsing!
OK      m4 requires 0.0, found 1.4.6

OK      make requires 3.79, found 3.81

OK      perl requires 5.6, found 5.10.0

Apple/Darwin ==> sed changed to gsed!
INSTALL gsed not found
        URL: http://www.gnu.org/directory/sed.html

macdev:hercules-3.08 hercules$
```

**Figure 144: Find missing packages**

In general everything looks fine with Mac OS X. If not, install the updates to the system to complete the prerequisites to configure and build the binaries. Do not install or update the "gawk" or "perl" package, a "gettext" and "sed" update is not necessary too.

Updates are available from the GNU Project (a Free Software Foundation project), have a look at the GNU Project web site http://www.gnu.org.

The updates are installed in the same manner like Hercules, as usual for GNU packages.

```
$ ./configure --prefix=/usr/local
$ make
$ sudo make install
```

**Note:** Do not install updates to a Mac OS X directory or replace a Mac OS X library.

On the other hand MacPorts and Homebrew are nice tools to keep your Mac OS X up-to-date too. To get more information how to do this, keep a closer look at the MacPorts web site http://www.macports.org  or the Homebrew web site http://mxcl.github.com/homebrew/.

## 22.4.3  The Build Process

Issue the following command to configure Hercules for your system.

```
$ ./configure --enable-setuid-hercifc=yes
```

By default, the configure script will attempt to guess appropriate compiler optimization flags for your system. If its guesses turn out to be wrong, you can disable all optimization by passing the "--disable-optimization" option to configure or specify your own optimization flags with the "--enable-optimization" option.

For additional configuration options, run "./configure –help"

Do not use "—prefix" with "/usr", this is a Mac OS X directory, use "/usr/local" instead (this is the default option).

**Note:** In the following examples "./configure --disable-nls --enable-setuid-hercifc" or "./configure --disable-nls --enable-setuid-hercifc --enable-optimization="-O3 -fomit-frame-pointer"" have been used to configure Hercules for a Mac OS X system on a PPC PowerBook and a Intel MacBook or MacMini.

Use "./configure with --enable-external-gui" to make GUI support available for applications like Hercules Studio GUI.
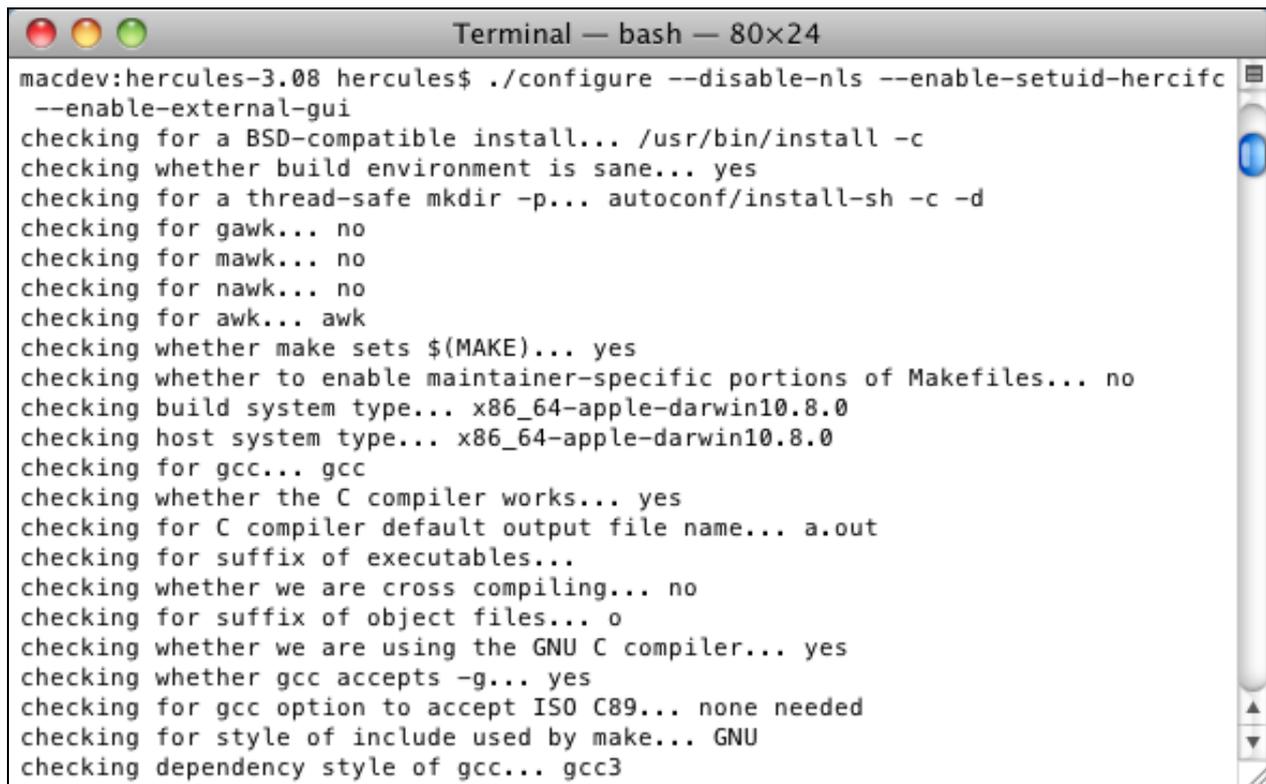
```
000                    Terminal — bash — 80×24
macdev:hercules-3.08 hercules$ ./configure --disable-nls --enable-setuid-hercifc
 --enable-external-gui
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... autoconf/install-sh -c -d
checking for gawk... no
checking for mawk... no
checking for nawk... no
checking for awk... awk
checking whether make sets $(MAKE)... yes
checking whether to enable maintainer-specific portions of Makefiles... no
checking build system type... x86_64-apple-darwin10.8.0
checking host system type... x86_64-apple-darwin10.8.0
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking for style of include used by make... GNU
checking dependency style of gcc... gcc3
```

**Figure 145: Configure the install process**
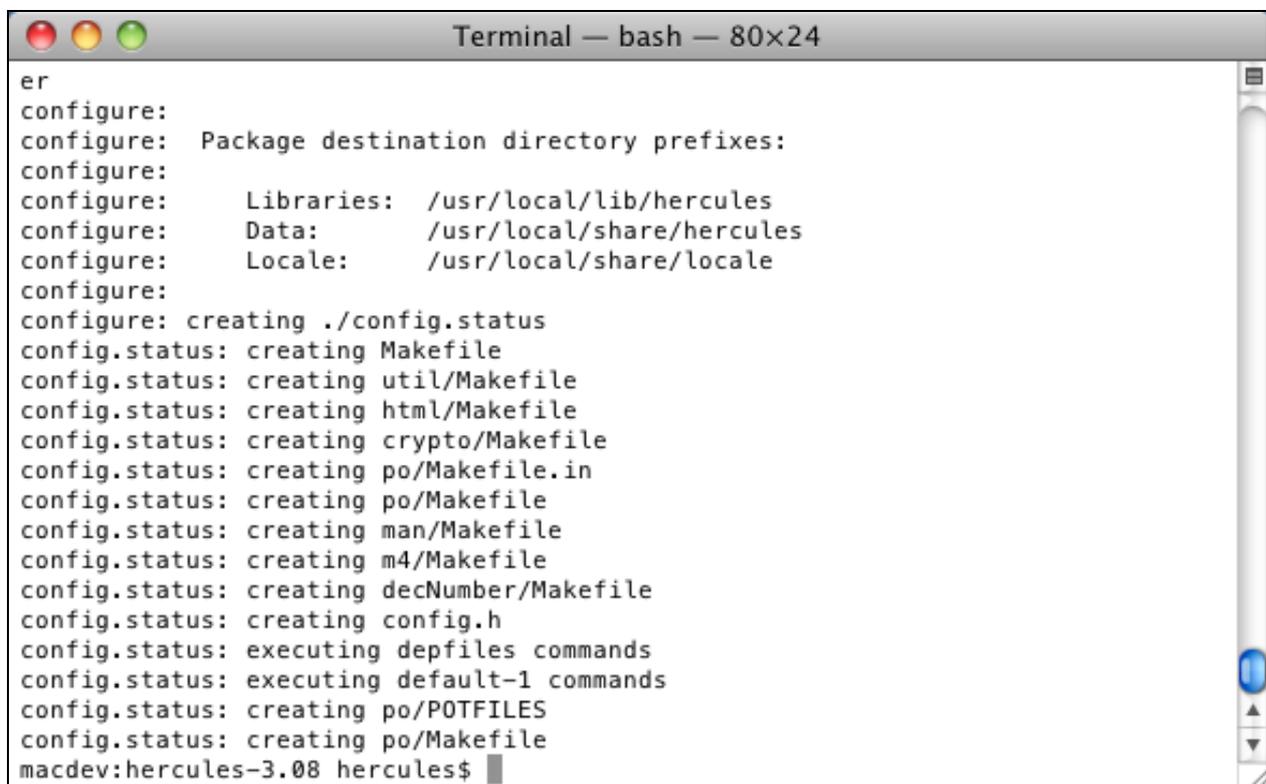
```
000                    Terminal — bash — 80×24
er
configure:
configure:   Package destination directory prefixes:
configure:
configure:      Libraries:   /usr/local/lib/hercules
configure:      Data:        /usr/local/share/hercules
configure:      Locale:      /usr/local/share/locale
configure:
configure: creating ./config.status
config.status: creating Makefile
config.status: creating util/Makefile
config.status: creating html/Makefile
config.status: creating crypto/Makefile
config.status: creating po/Makefile.in
config.status: creating po/Makefile
config.status: creating man/Makefile
config.status: creating m4/Makefile
config.status: creating decNumber/Makefile
config.status: creating config.h
config.status: executing depfiles commands
config.status: executing default-1 commands
config.status: creating po/POTFILES
config.status: creating po/Makefile
macdev:hercules-3.08 hercules$
```

**Figure 146: Create the makefiles**

If everything works well, the "configure" command ends with creating a lot of Makefile files and the corresponding messages.

Now issue the following command to build the executables.

```
$ make
```

Some minutes later, the "make" command should complete, without error messages similar to those shown in this snippet, some warning messages are normal.



**Figure 147: Make the executables**

The final step of the build process is to link the compiled object files into their binary forms and install them into our target directory.

Issue the following command to install the executables.

```
$ sudo make install
```

We will run the "make install" command under root authority so that we can be sure that any required folder can be created. The command should finish without error and the final messages will be similar to those shown in this snippet. Switch to a user with local administrator authority first.

---

**Figure 148: Make install the executables**

"hercifc" get the appropriate attributes if you use configure with option "--enable-setuid-hercifc", see chapter 22.6 "Configuring the network interface" for more information.

**Figure 149: Change the attributes**

The Hercules binaries are now installed, so you can skip to chapter 25 "Installation Verification Procedure" to continue.

By the way, the Hercules Emulator is available via the MacPorts and Homebrew package management software too. Have a look at the MacPorts web site http://www.macports.org or the Homebrew web site to get a list of the available packages.

With the MacPorts Profile and the Homebrew Formula some dependent libraries are installed too. But the configure option --enable-setuid-hercifc is not used.

MacPorts and Homebrew install packages to their own directory structure. MacPorts use "/opt/local" by default, but unfortunately Homebrew symlinks their files into "/usr/local". In general you can install your own stuff to "/usr/local" too, but be aware that if you install some libraries and tools yourself, it may cause some trouble when trying to build certain Homebrew Formula. As a result the Homebrew Doctor will warn you about this.

## 22.5 Verifying the Hercules installation

To verify the Hercules binaries installation open a new Terminal application and run the following Hercules command to show the Hercules version.

```
$ hercules 2>&1 | grep -i Version
```

**Figure 150: Display the Hercules version**

The command, like all BSD or Unix commands, is case sensitive. It starts the Hercules program and combines the two resulting output streams together ("2>&1") before piping that through a filter ("grep") searching for the text string "Version". As a result you get something like this:

```
Hercules Version v.rr
```

"v.rr" specifies the version and release of the Hercules Emulator.

But, if you get some messages similar to the following, some old libraries are not replaced correctly or another copy of Hercules was installed in the past:

```
HHCHD010I Dependency check failed for HERCULES, version(3.07) expected(3.08)

HHCHD010I Dependency check failed for REGS, version(3.03) expected(3.08)

HHCHD010I Dependency check failed for DEVBLK, version(3.05) expected(3.08)
```

**Note:** Check the MODPATH path in your Hercules configuration file and have a look at the shell environment variable "DYLD_LIBRARY_PATH".

## 22.6 Configuring the network interface

Once the Hercules binaries have been installed and verified, the following additional tasks should be completed before starting to configure your new Hercules environment.

In order to support IP routing to a Hercules Guest system, a network tunnel is created using the native Mac OS X "tunx" device. This device simulates a network layer 3 device and routes IP packets between the Mac OS X operating system and Hercules.

The definition of a 3088 CTC device in the Hercules configuration file will make Hercules call a special program specifically to start and configure the "tunx" device.

Alternatively a network link layer device is created using the native Mac OS X "tapx" device. This device simulates a network layer 2 device and routes ethernet frames between the Mac OS X operating system and Hercules.

The definition of a 3172 / OSA LCS device in the Hercules configuration file will make Hercules call a special program specifically to start and configure the "tapx" device too.

The tunnelling device (tunx) or link layer device (tapx) is not active (as a configurable device object) until it is opened by Hercules.

The program to start and configure is called "hercifc" and is installed as part of the Hercules binaries. As we are not running Hercules with root authority and this network device program will require root authority to issue system calls, we need to run some additional commands to allow it to work correctly.

**Note:** If you use option --enable-setuid-hercifc with "yes", configure adds the necessary commands to do this to the make install process. You can use a group name instead of "yes" to change the group.

Issue the following commands in the Terminal application window.

Change to the directory containing the "hercifc" program.

```
$ cd /usr/local/bin
```

List the current attributes of the hercifc program.

```
$ ls -la hercifc
```

Use "sudo" to change the ownership of the "hercifc" program to user "root" and group "admin".

```
$ sudo chown root:admin hercifc
```

Use "sudo" to change the attributes of the "hercifc" program to be setuid to the owner ("root") and executable only by root and the "admin" group. Switch to a user with local administrator authority first.

```
$ sudo chmod 6750 hercifc
```

**Figure 151: Configure the Network Interface**

The result of these commands is to allow the "hercifc" program to execute with root authority (that is what the chmod command above does) but restrict the execution of the program only to the file owner (user "root") and any user connected to the "admin" group.

This will allow the tunnelling device to open and configure correctly when initiated by user "hercules" but reduce the security exposure of having a program setuid-to-root to the minimum.

**Note:** We should use the Administrators Group "admin" or the System Group "wheel". On the other hand, with the Users Group "staff" everyone is authorised to open the tunnelling device.

Finally use "dseditgroup" to assign the Hercules user to the corresponding group.

```
$ sudo dseditgroup -o edit -a hercules -t user admin
```

Otherwise, you can define the Hercules user as a local administrator.

To reflect to the Hercules package file use "dseditgroup" to assign the Hercules user to the "wheel" group. This is the preferred way.

```
$ sudo dseditgroup -o edit -a hercules -t user wheel
```

```
macdev:~ admin$ id hercules
uid=507(hercules) gid=20(staff) groups=20(staff),402(com.apple.sharepoint.group.
1),61(localaccounts),12(everyone),403(com.apple.sharepoint.group.2),404(com.appl
e.sharepoint.group.3)
macdev:~ admin$ sudo dseditgroup -o edit -a hercules -t user wheel
Password:
macdev:~ admin$ id hercules
uid=507(hercules) gid=20(staff) groups=20(staff),402(com.apple.sharepoint.group.
1),61(localaccounts),12(everyone),0(wheel),403(com.apple.sharepoint.group.2),404
(com.apple.sharepoint.group.3)
macdev:~ admin$ 
```

**Figure 152: Assign to group wheel**

# 23. Installing additional software

## 23.1 Installing the tun/tap driver

You need the Tun/Tap driver to provide networking from your Hercules Guest system.

### 23.1.1 Downloading the binaries

The ready-to-run binaries can be downloaded from http://tuntaposx.sourceforge.net. For Mac OS X users there is currently one installation disk image available.

The installation package (http://downloads.sourceforge.net/tuntaposx/tuntap_20111101.tar.gz) should work with 10.6 (Snow Leopard) and above. The source code to build the Tun/Tap driver is also available.

The Hercules installation disk images contain an older Tun/Tap Driver installation package.

### 23.1.2 Installation steps

To start the installation dialog, unpack the ".tar.gz" archive file and double click the ".pkg" package file. As an alternative you can open the Hercules installation disk image ".dmg" file again and switch to the tuntap folder and double click the ".pkg" package file. Tun/Tap use less than 1 MB of disk space of your system disk "Macintosh HD". First a welcome window is presented.
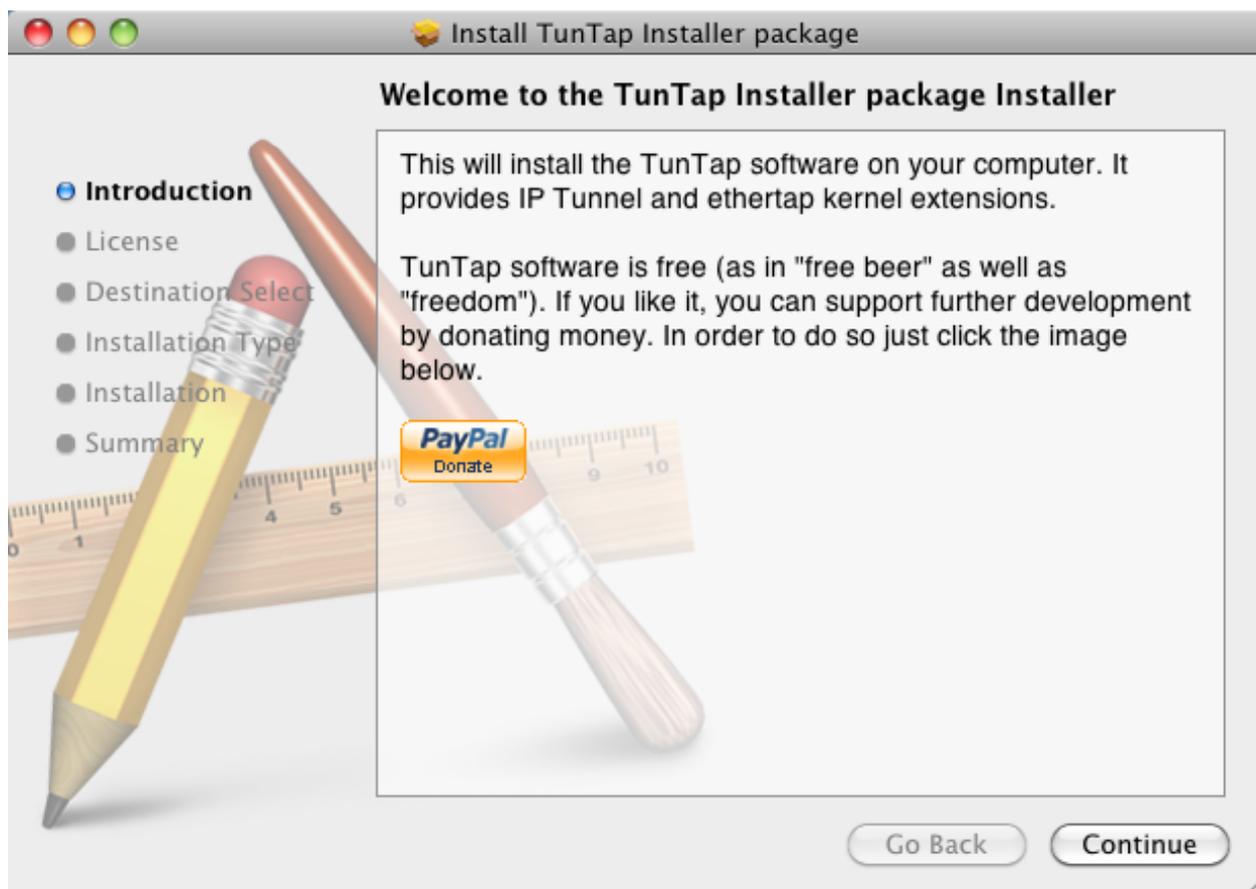
**Figure 153: Tun/Tap welcome window**

Click on "Continue" twice to continue the installation process then click "Agree" to accept the licence agreement.

Click to "Install" and then enter the name for an administrator and the password and click "OK" to perform the standard installation process. The installer now begins to copy files to the system disk "Macintosh HD".

After a few seconds the installation process will finish and present the final window. Click on "Close" to terminate the installation process.

## 23.1.3 IP Forwarding

You may or may not need to have "IP Forwarding" enabled on your Mac OS X system. To allow your Hercules guest operating system to communicate with hosts on the LAN apart from the host machine itself (where Hercules is running), TCP/IP requires routes to and from the Hercules Guest operating system.

This is necessary so that the Hercules Guest packets can be properly routed to their final destination. This is typically the role performed by a hardware router. If you do not use a hardware router then Mac OS X "IP Forwarding", together with appropriate route statements, will perform this role.

Add the following to the "mvs38.csh" start-up script before the Hercules command, see "Shell start-up script".

```
#
ifconfig en0 alias 192.168.200.254
#
sysctl -w net.inet.ip.forwarding=1
#
# ifconfig tun0 192.168.200.254 192.168.200.20 netmask 255.255.255.0
# ifconfig tun0 up
# ifconfig tap0 192.168.201.0 netmask 255.255.255.0
# ifconfig tap0 up
(sleep 5; route add -net 192.168.201.0 -interface tap0 &>> log/start.log) &
#
```

**Note:** Use "sudo" to start the start-up script.

## 23.1.4 Configuring Hercules

Two IP addresses must be assigned, one for the Hercules end of the link and another for the guest system end of the link. Add the following to your configuration file.

```
#
# CTCI Device
0400.2 CTCI -n /dev/tun0 -s 255.255.255.255 192.168.200.20 192.168.200.254
#
# OSA Devices
0440.2  LCS -n /dev/tap0 192.168.201.20
#
```

The first IP addresses (192.168.200.20 and 192.168.201.20) are the IP addresses of your Hercules guest system, the operating system running under Hercules. The second IP address (192.168.200.254) itself identifies the host machine (where Hercules is running).

Read the "README.NETWORKING" file included with the source code for additional and updated instructions to networking.

**Note:** The name of the virtual interfaces cannot be left unspecified in Mac OS X. The driver would not pick a name by trying to allocate the next free device of that kind. Consider a virtual interface cannot be used twice!

Details of the configuration file are found in the Hercules "User Reference Guide".

## 23.2  Installing the 3270 client

You need a 3270 client to get access to guest systems that are running guest operating systems like MVS 3.8, VM 6.0 or any other classic mainframe System/370 guest system.

### 23.2.1  Downloading the binaries

The ready-to-run binaries for the Brown TN3270 Terminal Emulation tn3270 X can be downloaded from http://www.brown.edu/cis/tn3270/. For Mac OS X users there are currently some installation disk images available.

The installation package (see http://www.brown.edu/cis/tn3270/tn3270_X_3.2.4.dmg) should work with Mac OS X 10.6 (Snow Leopard) and below, the newer installation package (see http://www.brown.edu/cis/tn3270/tn3270_X_3.3b6-cs.dmg) should work with Mac OS X 10.6 (Snow Leopard) and above.

### 23.2.2  Installation steps

Open the Applications folder at the root level of your hard drive and create a new folder named "TN3270 X". Then open the disk image and copy the tn3270 X application and the other files and folder to the new folder in the Application folder.

tn3270 X uses 6 MB of disk space of your system disk "Macintosh HD" or any other disk where you copy the tn3270 X application.
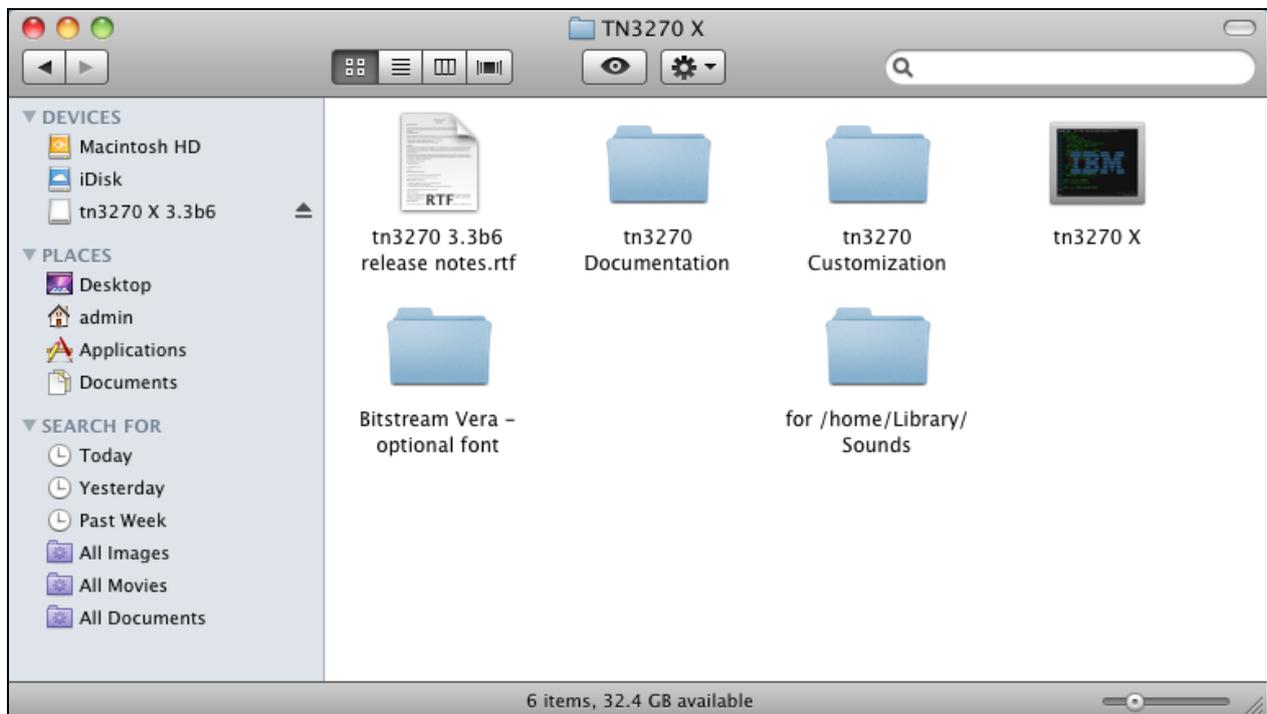


**Figure 154: TN3270 X folder**

### 23.2.3 Customization steps

There is one setting that must be changed to use the terminal emulation with some operating systems, especially for MVS 3.8 for System/370. The first time open a connection to Hercules before IPL'ing the system, go to the "Session", "Features" menu and set "Change embedded nulls to blanks" to "No". Then click on "OK". Now click on "File", "Save default settings..." to make the settings permanent.
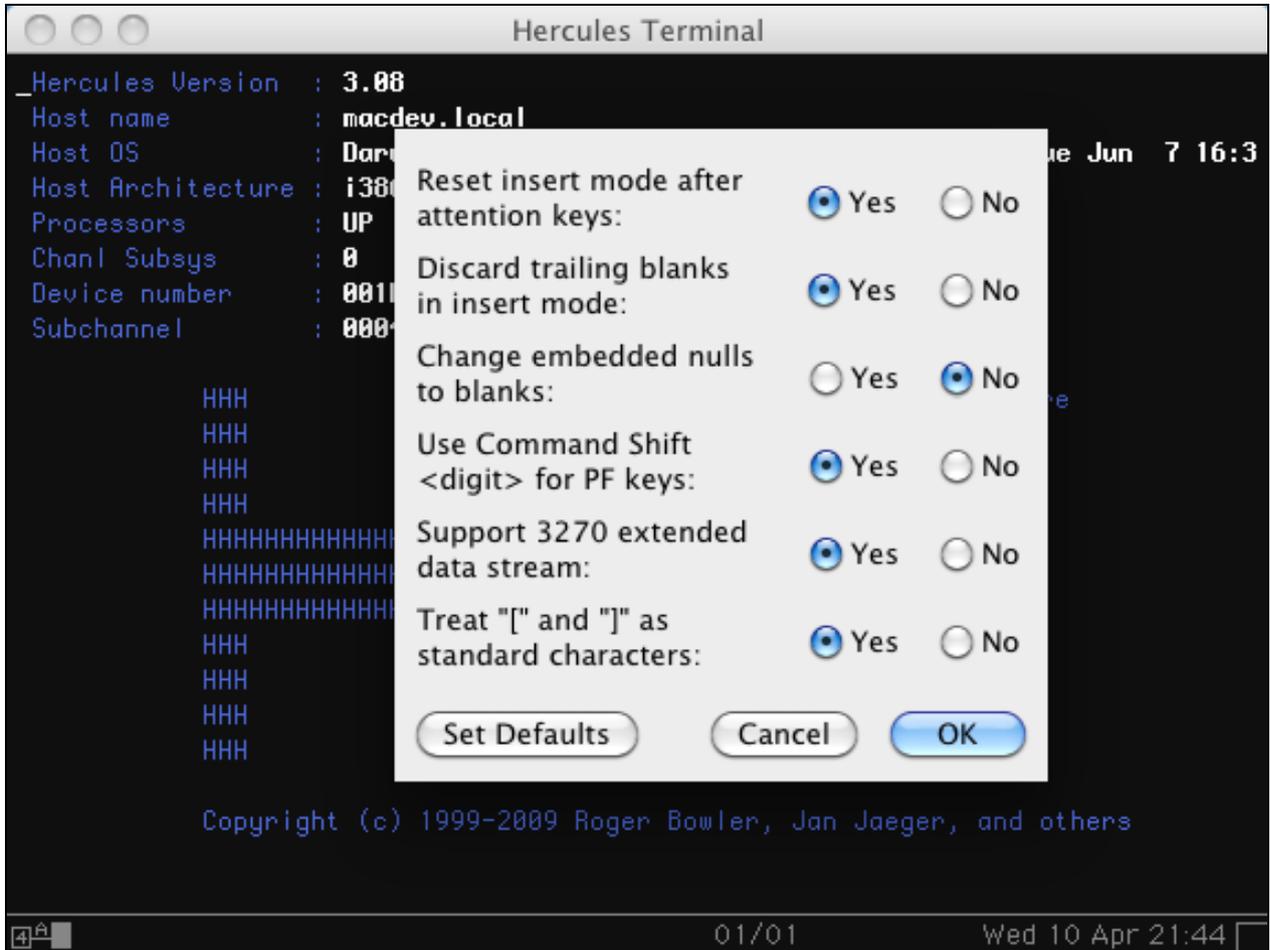


**Figure 155: TN3270 X terminal settings**

Unfortunately the 3270 Client tn3270 X was shipped with a limited set of keyboard mappings. In the tn3270 customization folder are some short manuals available to give you some help to customize the tn3270 X application.

The keyboard customization folder contains the built-in keyboard mappings and a sample Austrian and German keyboard mapping. To add your preferred keyboard mapping, modify one of the shipped built-in keyboard mapping files. Change the "name:" operand to your own keyboard name otherwise you replace an existing keyboard mapping.

**Figure 156: German keyboard mapping**

To get more information about the key codes used, start the KeyMap application from the keyboard customization folder and enable "Display Keystroke Information" in the "File" menu.

Choose "Open definition file" from the "File" menu, and open your keyboard mapping file. Then choose "Write resource to application package" from the "File" menu. When the standard file dialog appears, choose the tn3270 X application to update.

**Note:** The current version of the KeyMap application does not provide a method for deleting keyboard mappings from tn3270 X, but adding and replacing keyboard mappings will usually be sufficient.

## 23.2.4 Building from source

In order to build a 3270 Client have a look at the web site http://sourceforge.net/projects/x3270/ where you get the Suite 3270 source code from. Download and unpack the source code and build the c3270, s3270 and pr3287 in the same manner like Hercules.

The used Suite 3270 uses less than 2 MB of disk space of your system disk "Macintosh HD". Additional information is available from web site http://x3270.bgp.nu.

```
$ ./configure --without-readline --disable-nls --disable-dbcs

$ make

$ sudo make install
```

For additional configuration options, run "./configure –help"

**Note:** Do not install the commands into a Mac OS X directory, use --prefix with "/usr/local" (this is also the default option).

If everything works well, you can start the 3270 Client or the 3287 Printer from the terminal application to verify the installation. It is assumed that Hercules is already running.

```
$ c3270 localhost:3270
```



**Figure 157: C3270 terminal**

MacPorts and Homebrew are not only nice tools to keep your Mac OS X up-to-date, you can also install tools like the Suite 3270 with MacPorts or Homebrew. To get more information how to use MacPorts or

Homebrew, keep a closer look at the MacPorts web site http://www.macports.org or the Homebrew web site http://mxcl.github.com/homebrew/.

## 23.3 Installing the Hercules Studio GUI

The Hercules Studio GUI provides a graphical user interface under Mac OS X to the Hercules Emulator itself. Hercules itself has only a semi-graphical user interface, the Hercules Studio GUI provides a full graphical user interface.

You cannot use Hercules Studio GUI with the ready-to-run Hercules Emulator binaries up to version 3.07. You must build the Hercules Emulator yourself, see "Building from Source" to get more information. Beginning with Hercules version 3.08 GUI support is built-in by default for all platforms.

### 23.3.1 Downloading the binaries

The ready-to-run binaries can be downloaded from http://www.jacobdekel.com/hercstudio/. For Mac OS X users there is currently one installation disk image available.

The installation package (see http://www.jacobdekel.com/hercstudio/hercstudio-1.4.0-snowleopard.dmg) should work with Mac OS X 10.6 (Snow Leopard) and above. The used Qt runtime libraries are included now.

### 23.3.2 Installation steps

Open the applications folder at the root level of your hard drive. Then open the disk image and copy the Hercules Studio application to the application folder.

Hercules Studio GUI uses 38 MB of disk space of your system disk "Macintosh HD" or any other disk where you copy the Hercules Studio application to.

### 23.3.3 Running the Hercules Studio GUI

To start the Hercules Studio GUI start the Hercules Studio application. If you install the Hercules Emulator to the default directory structure, Hercules Studio GUI will become a graphical user interface for Hercules.
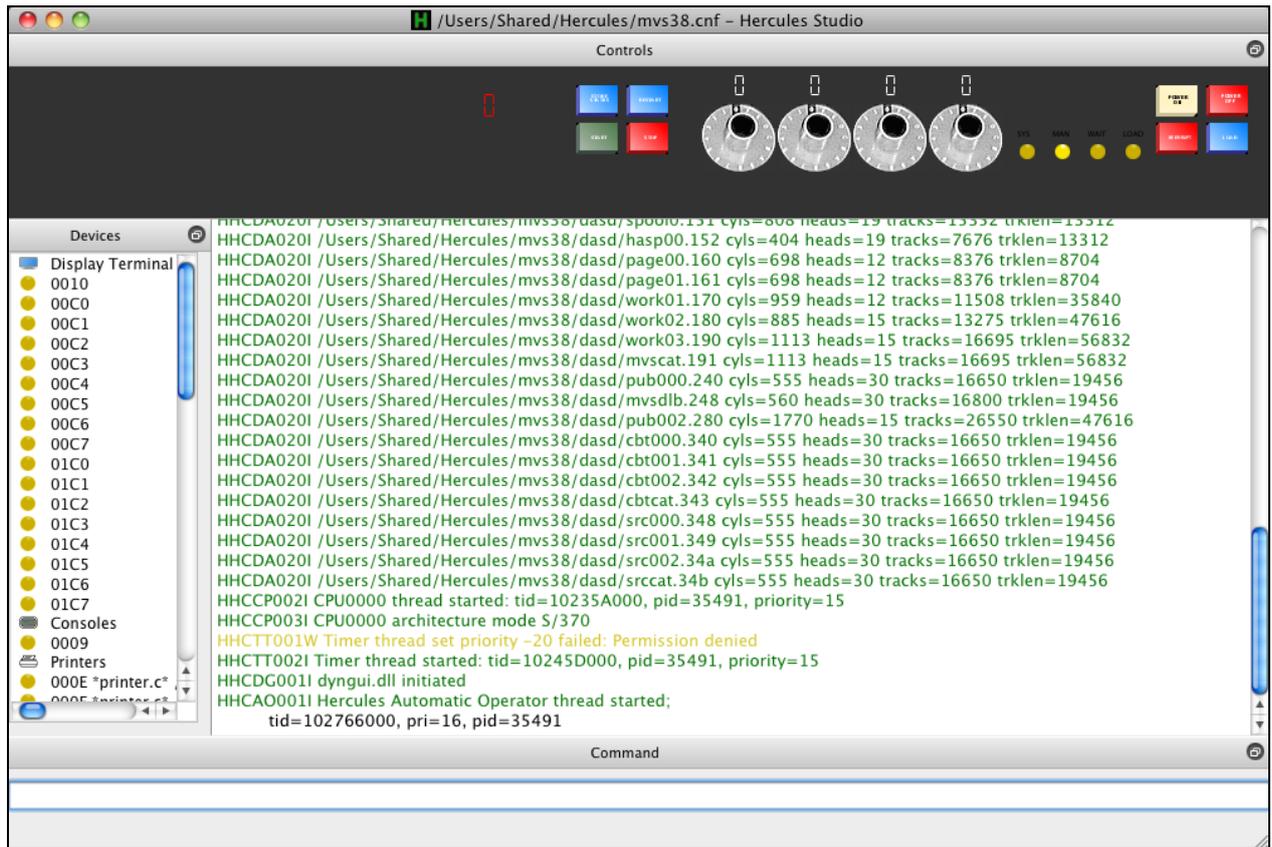
**Figure 158: Hercules Studio GUI**

# 24. Customization steps

After the actual installation is completed there are some additional customization steps required.

- Creating the Hercules guest folder.
- Creating the Hercules configuration file.
- Creating a Hercules guest start-up script.
- Creating a Hercules run-commands file.
- Creating a Hercules console link file.

These manual customization steps are explained in a short form during the following sections. The details are explained in the Hercules "User Reference Guide".

## 24.1 Creating the Hercules guest folder

We are going to create some script and configuration files and we need to identify where we need to create them now. Additionally, we will need to identify a location for our guest operating system, for example MVS 3.8 for System/370.

Use a completely separate directory for the Hercules Guest operating system.

| | |
|---|---|
| `/Users/Hercules` | Hercules user home directory. |
| `/Users/Shared/Hercules` | Guest operating system files. |
| `/mvs38` | Guest configuration file and the run-command script file. |
| `/jcl` | JCL files |
| `/dasd` | DASD files |
| `/tape` | Tape files |
| `/prt` | Printer files |
| `/rdr` | Reader files |
| `/pch` | Punch files |
| `/log` | Log files |
| `/tools` | Various tools |

**Figure 159: Mac OS X Directory Structure Example**

**Note:** Exclude the guest operating system from Time Machine to prevent problems with your guest and to save space on your backup media.

The Mac OS X file system, in contrast to BSD or Unix file systems, is not case sensitive.

## 24.2 Creating the Hercules configuration file

When starting the Hercules Emulator from either the terminal application or via a launcher, you must specify the name of a configuration file as a parameter.

```
$ Hercules -f filename
```

"*filename*" is the name of the configuration file. The default filename is "hercules.cnf" if none is specified during the start-up.

The configuration file is an ASCII text file that is used to describe the processor definition, the device layout and any runtime parameters. Details can be found in the Hercules "User Reference Guide".

Using the directory structure example outlined before, we create this file in the following folder.

```
/Users/Shared/Hercules/mvs38
```

## 24.3 Creating a Hercules guest start-up script

Although the Hercules Emulator can be started manually from the terminal application, it is often easier to establish a script file to do this. The script file can be executed by running it from the shell or starting it from the Finder.

Using the directory structure example outlined before, we create these files in the following folder.

```
/Users/Shared/Hercules
```
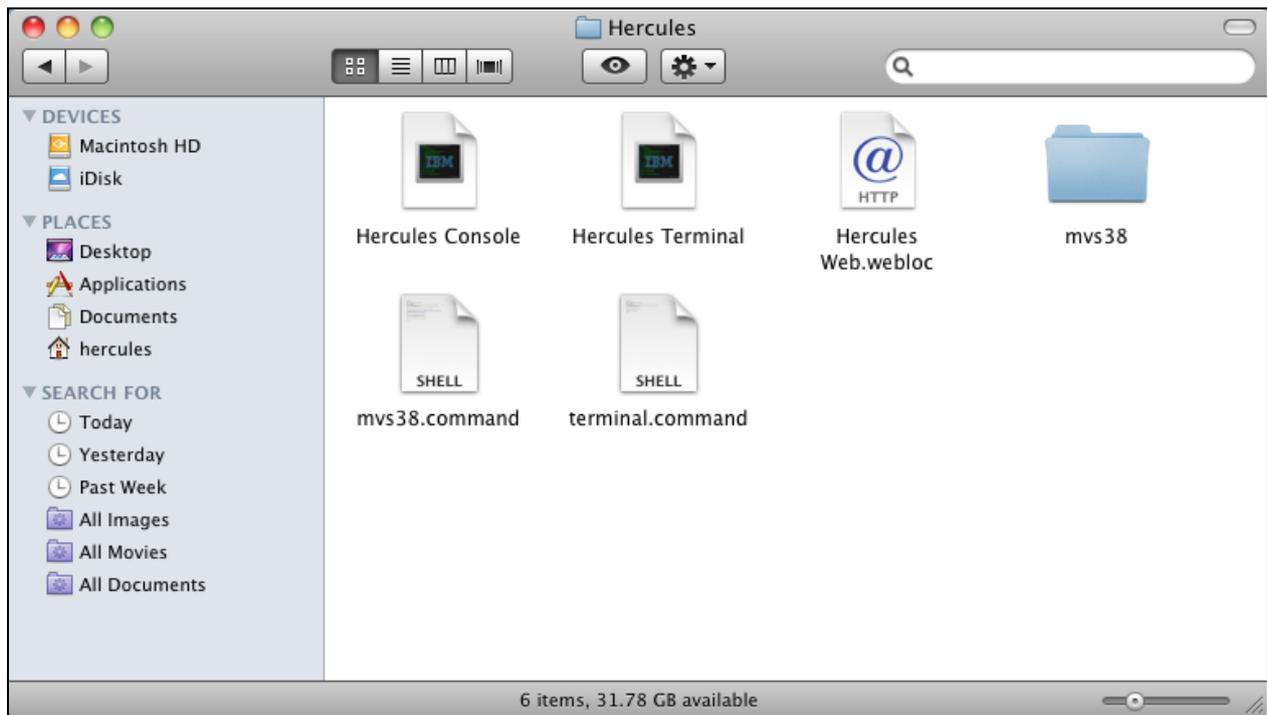
**Figure 160: The shared Hercules folder**

## 24.3.1 Finder launcher script

We should use the name "mvs38.command" for our Finder launcher script. The following shows an example of a Hercules start-up script file. You should use the TextEdit application or one of the installed editors (or "vi" if you feel adventurous).

```
#!/bin/tcsh

#setenv PATH "${PATH}:/usr/local/bin"

setenv DYLD_LIBRARY_PATH "/usr/local/lib/hercules"

rehash

#

cd /Users/Shared/Hercules/mvs38/

/usr/local/bin/hercules -f mvs38.cnf > log/hercules.log

#

exit
```

**Figure 161: Finder launcher script**

### 24.3.2 Shell startup script

We should use the name "mvs38.csh" for our startup script.

```
#!/bin/tcsh
#setenv PATH "${PATH}:/usr/local/bin"
setenv DYLD_LIBRARY_PATH "/usr/local/lib/hercules"
rehash
#
cd mvs38
/usr/local/bin/hercules -f mvs38.cnf > log/hercules.log
#
```

**Figure 162: Shell startup script**

### 24.3.3 Make the scripts executable

We have created our script files, now we need to make them executable. This is necessary as Mac OS X can execute a file only when it is marked to be executable. This is a security feature to try to reduce the chance of creating a dangerous command file by mistake.

We are going to issue four commands, start the Terminal application again.

```
$ cd /Users/Shared/Hercules
$ chmod 750 mvs38.command
$ chmod 750 mvs38.csh
$ exit
```

**Figure 163: Making the scripts executable**

## 24.4 Creating a Hercules run-commands file

Hercules also provides the ability of automatically executing Hercules panel commands after start-up. If a "hercules.rc" file exists when Hercules starts, each line contained in the file is read and interpreted as panel command.

Using the directory structure example outlined in "Creating the Hercules Guest folder", we create this file in the following folder.

```
/Users/Shared/Hercules/mvs38
```

We should use the name "hercules.rc" for our run-command script.

```
pause 3
# sh open "../Console" &
# pause 1
sh open "../Terminal" &
pause 1
sh open "../Terminal" &
# pause 1
# sh open "http://127.0.0.1:8081" &
pause 5
# ipl 148
```

**Note:** "Console" and "Terminal" are tn3270 X application configuration files. The last "open" starts the Safari application to connect to the Hercules HTTP server (commented out in this example).

## 24.5  Creating a Hercules console link file

Create a link file to connect to the Hercules HTTP server with the Safari application. Using the directory structure example outlined before, we create this file in the following folder.

```
/Users/Shared/Hercules
```

We use the name "hercules.webloc" for our console link. The three dots at the end of the second line in the console link file are to show a continuation on the following line. The dots itself must not be coded.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN" ...
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
      <key>URL</key>
      <string>http://127.0.0.1:8081/</string>
</dict>
</plist>
```

**Figure 164: Console link file**

# 25. Installation Verification Procedure

To check if everything works well, try to start the ZZSA Cards with the Hercules configuration shipped with the Hercules source code.

## 25.1 Start ZZSA

Open the Terminal application again and switch to your source code folder and start Hercules with the following command.

```
$ hercules
```



**Figure 165: Hercules Console Window**

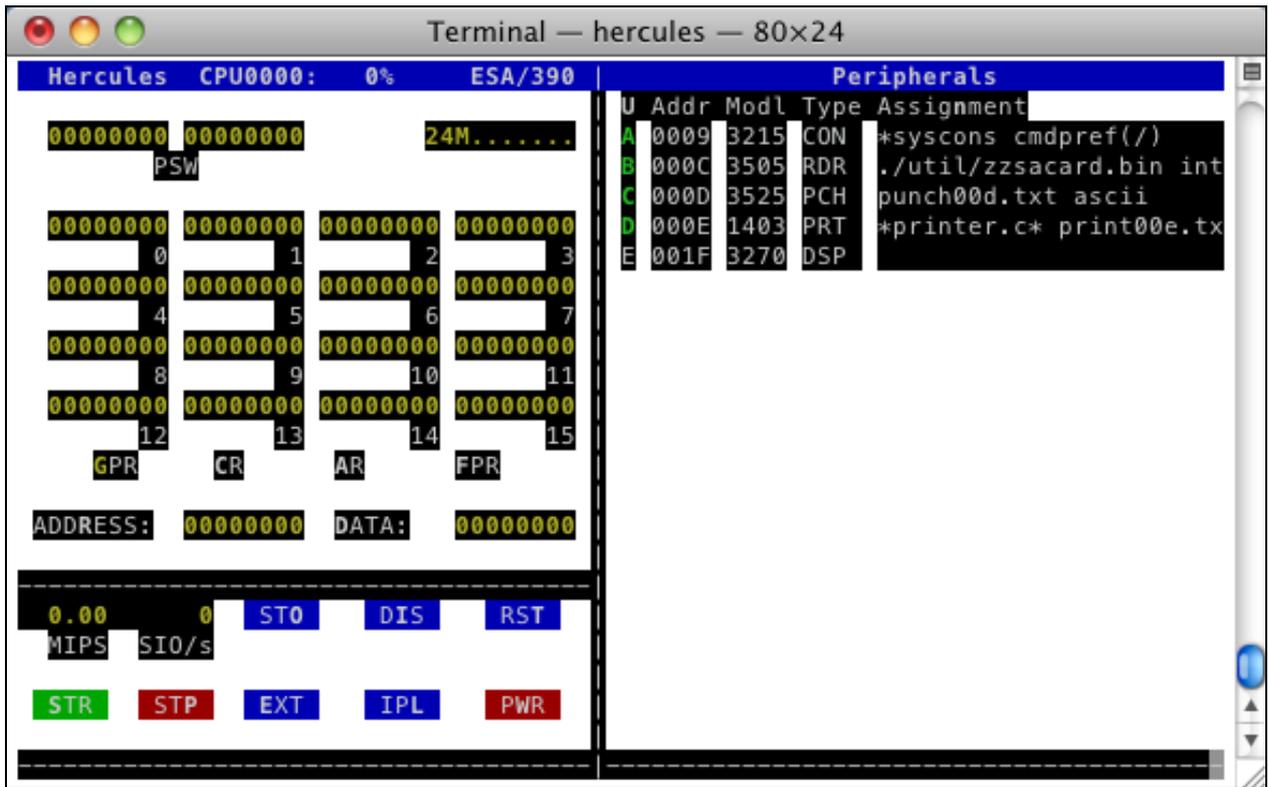Have a look at the second view of the Hercules console, switch to the second view with the Esc key and back again.

**Figure 166: Hercules Device and Status Display**

Start your preferred 3270 Client application and connect to Hercules (try "localhost" with port 3270).
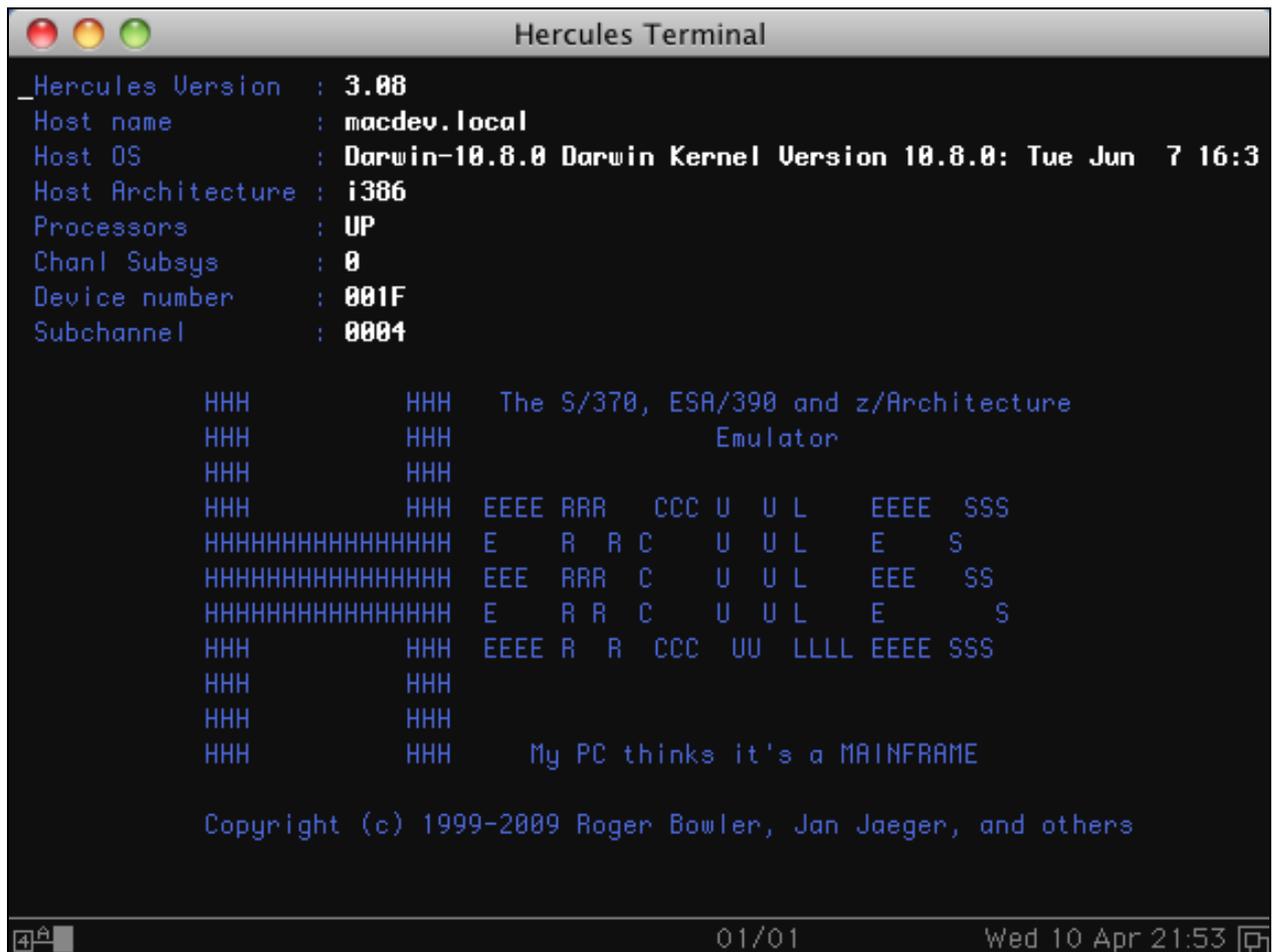
**Figure 167: TN3270 Terminal**

Start the Safari application and open the link to Hercules ("http://localhost:8081").
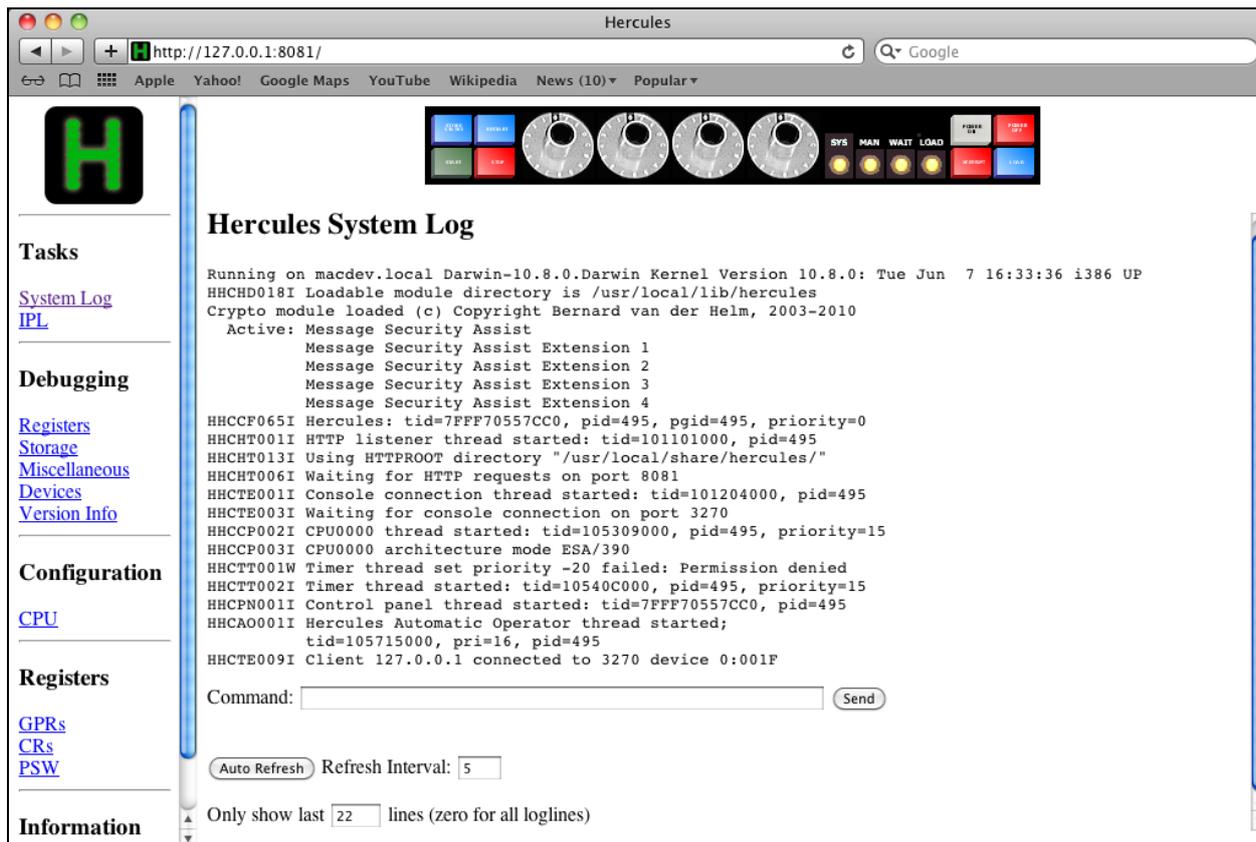
**Figure 168: Hercules Web Browser Interface**

Start (IPL) the ZZSA editor card deck from the 3505 device 000C on the Hercules console or from Safari.

If you are interested in using Hercules Studio GUI for testing purpose, it is necessary to change the ZZSA card file relative path to an absolute path name in the "hercules.cnf" sample file.

```
000C    3505    ./util/zzsacard.bin
000C    3505    /Users/hercules/Projects/hercules-3.08/util/zzsacard.bin
```

**Figure 169: Hercules Console IPL**

## 25.2  ZZSA Logon

Have a look at the 3270 client and press the "Clear Screen" or "Enter" key. The ZZSA password screen became available now, enter the password with "zzsecret" (the password will not be displayed as you type it) and press the "Enter" key.
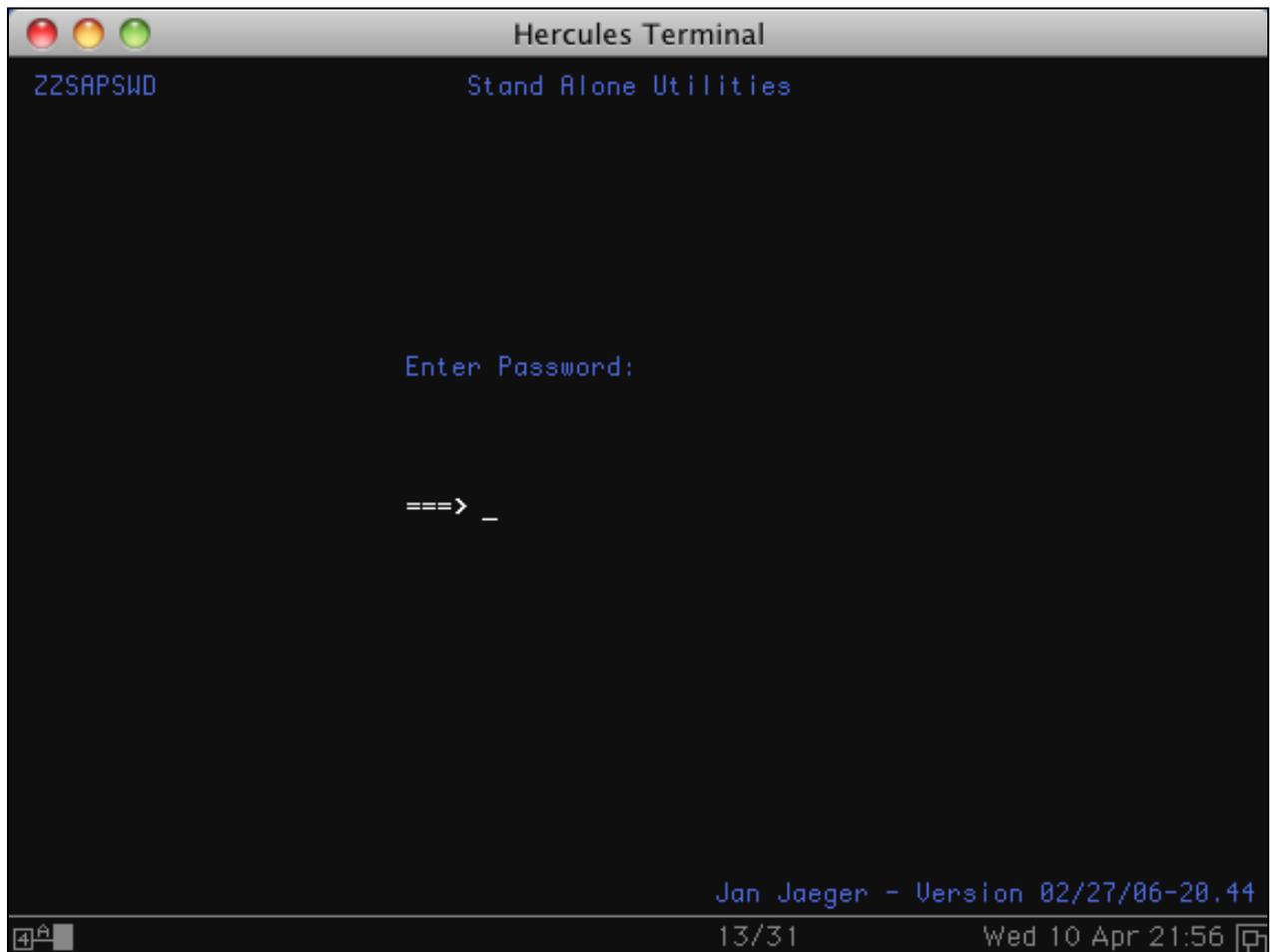
**Figure 170: ZZSA Password Screen**

Next the ZZSA Primary Menu becomes available.

**Figure 171: ZZSA Primary Menu**

Select option "0" and press the "Enter" key to get the ZZSA device list.

**Figure 172:ZZSA Device List**

Press "PF3" to end the current panel and going back to the previous screen again.

**Figure 173: ZZSA Primary Screen**

## 25.3 Quit ZZSA

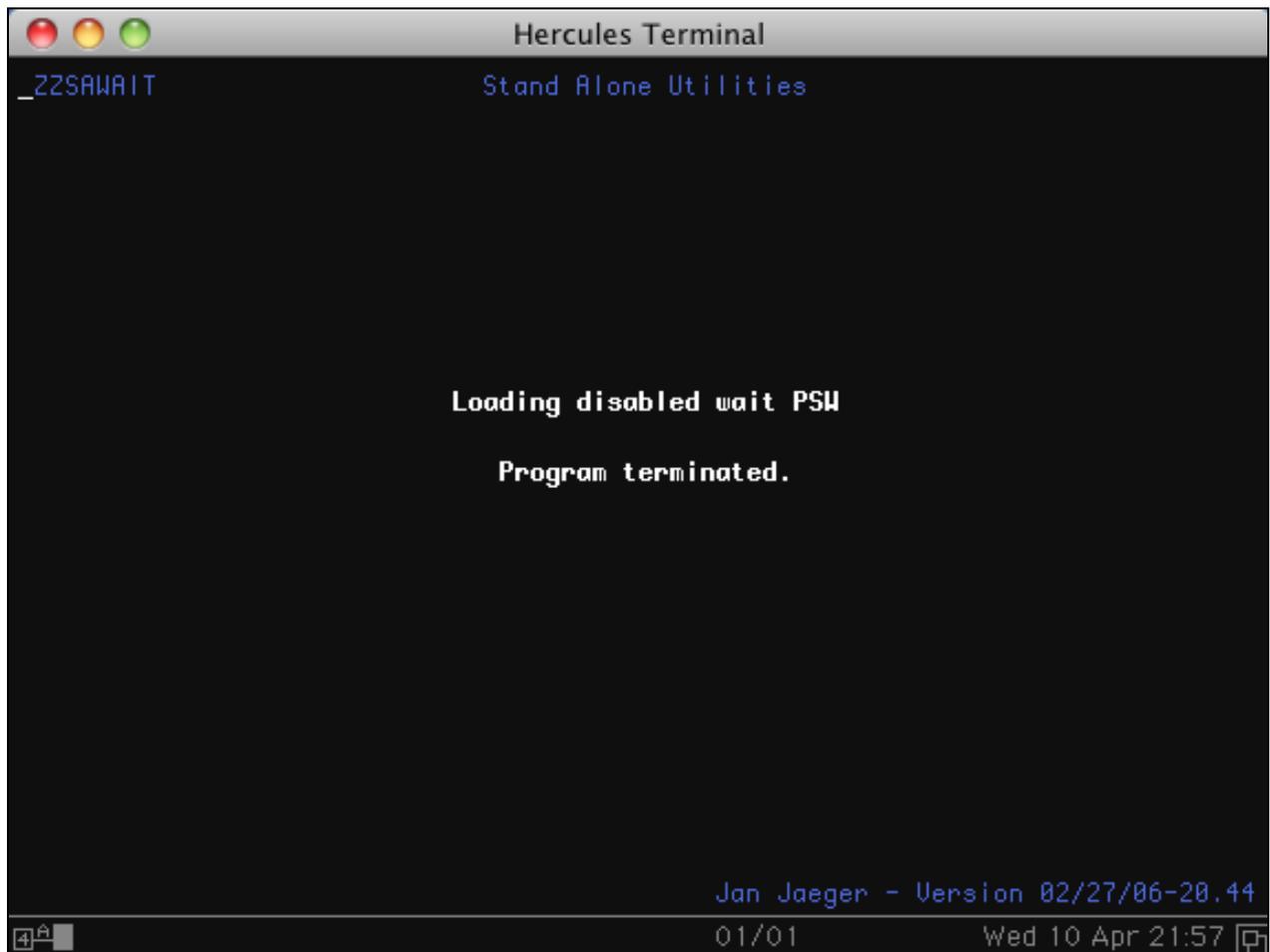Select option "X" and press the "Enter" key to terminate the ZZSA program.

**Figure 174: Quit ZZSA**

Finally type "QUIT" or "EXIT" on the Hercules console or from Safari to stop Hercules. Hercules will shut down now.

```
CR00=00000840   CR01=00000000   CR02=00000000   CR03=00000000
CR04=00000000   CR05=00000000   CR06=80000000   CR07=00000000
CR08=00000000   CR09=00000000   CR10=00000000   CR11=00000000
CR12=00000000   CR13=00000000   CR14=C2000000   CR15=00000000
HHCCP014I CPU0000: Operation exception CODE=0001 ILC=2
PSW=00082000 8000006A INST=000A          ????? ,                    ?
GR00=00010004   GR01=000083A8   GR02=01000068   GR03=00000000
GR04=00000000   GR05=00000000   GR06=000005B8   GR07=00000000
GR08=00000000   GR09=00000000   GR10=00001000   GR11=00002000
GR12=00000000   GR13=00000000   GR14=80000390   GR15=00002920
HHCCP011I CPU0000: Disabled wait state
          PSW=000A0000 DEAD0000
quit
HHCIN900I Begin Hercules shutdown
HHCIN901I Releasing configuration
HHCTE004I Console connection thread terminated
HHCHD902I logger_term complete
HHCHD909I Shutdown sequence complete
HHCIN904I All termination routines complete
HHCIN909I Hercules shutdown complete
HHCIN099I Hercules terminated
HHCHD900I Begin shutdown sequence
HHCHD909I Shutdown sequence complete
macdev:hercules-3.08 hercules$ █
```

**Figure 175: Hercules Shutdown**

If you successfully followed the previous procedures then it can be assumed that Hercules is working correctly.

# Appendix A. Links

- **The Hercules System/370, ESA/390, and z/Architecture Emulator**

    http://www.hercules-390.eu

- **Hercules source code repositories**

    https://github.com/rbowler/spinhawk        (release 3.xx development stream)

    https://github.com/rbowler/sandhawk        (release 4.xx development stream)

    https://github.com/s390guy/hercules-390        (cutting-edge developer sandbox)

- **Hercules Developer Snapshots (Dave Wade)**

    http://www.smrcc.org.uk/members/g4ugm/snapshots/

- **Hercules PDF Documentation (Peter Glanzmann)**

    http://hercdoc.glanzmann.org

- **The MVS Tur(n)key System, Version 3 (Volker Bandke)**

    http://www.bsp-gmbh.com/turnkey/index.html

- **Hercules WinGUI ("Fish", David B. Trout)**

    http://www.softdevlabs.com/Hercules/hercgui-index.html

- **CTCI-W32 ("Fish", David B. Trout)**

    http://www.softdevlabs.com/Hercules/ctci-w32-index.html

- **Hercules Studio (Jacob Dekel)**

    http://www.mvsdasd.org/hercstudio

- **Hebe – Hercules Image Manager (Robin Atwood)**

  http://kde-apps.org/content/show.php/Hebe?content=126738

- **WinPcap, Politecnico di Torino**

  http://www.winpcap.org

- **Vista tn3270, Tom Brennan Software**

  http://www.tombrennansoftware.com

- **X3270, Paul Mattes**

  http://x3270.bgp.nu

- **AWSBROWSE ("Fish", David B. Trout)**

  http://www.softdevlabs.com/Hercules/hercgui-index.html

- **XMIT Manager**

  www.cbttape.org

- **CBT MVS Utilities Tape (CBTTAPE)**

  www.cbttape.org

- **Microsoft Visual C++ 2008 Express**

  http://www.microsoft.com/express/download/

- **ZLIB**

  http://www.zlib.net

  http://www.softdevlabs.com/Hercules/ZLIB1-1.2.3-bin-lib-inc-vc2008-x86-x64.zip


- **BZIP2**

  http://www.bzip.org

  http://www.softdevlabs.com/Hercules/BZIP2-1.0.5-bin-lib-inc-vc2008-x86-x64.zip


- **PCRE**

  http://www.pcre.org

  http://www.softdevlabs.com/Hercules/PCRE-6.4.1-bin-lib-inc-vc2008-x86-x64.zip

Hercules Emulator

Hercules System/370, ESA/390,
z/Architecture Emulator

# Installation Guide

Version 3 Release 09

HEIG030900-00